School of Engineering and Science
Department of Electrical and Computer Engineering

3D Point Cloud Representation Learning and Reconstruction using Vector-based Neural Network

Jonathan Phang Then Sien

0000-0002-4797-7375

**This thesis is presented for the Degree of
Doctor of Philosophy
of
Curtin University**

March 2022

Declaration

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgment has been made.

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

Signature:
Date:

Acknowledgements

First and foremost, I would like to express my gratitude to my principal supervisor A/Prof. King Hann Lim for his continuous guidance and encouragement throughout my studies, especially during the pandemic period. He literally turned me from nobody to somebody in regards to this field of study and think beyond the confined perspective. I have learnt not only new knowledge but his dedication, passion and determination that drive me to the completion of my thesis. I appreciate all the time and efforts he spent to improve the quality of my research.

Abstract

Three-dimensional (3D) point cloud representation learning and reconstruction using Deep Learning (DL) architecture has become an active research trend due to the rapid development of affordable and accessible 3D acquisition technologies such as laser scanners, RGB-D cameras, and stereo vision cameras. In many representations, point cloud has received the most interest and favoured in many research works due to its simplistic and flexible data structure in three dimension space. Recent research works deploy DL to process raw point clouds by taking the advantages of its learning capacity to overcome challenges in point clouds such as incomplete, noisy, and unstructured in gridless space. Consequently, deployment of DL in point cloud representation learning and reconstruction have led to many real-world applications such as autonomous robotic navigation, object classification and 3D modeling.

Learning representation of a point cloud using DL poses challenges due to sparsity and its un-ordered structure. Presently, conventional point cloud models with single latent feature representation are not sufficient to fully capture the complex geometry of a point cloud. Hence, Point cloud Neural Density Estimator (PNDE) is proposed as a parametric mixture model to produce multiple local representations of a point cloud, where each representation is a density parameter (Gaussian components), i.e. means and variances. In quantitative evaluations, classification performance of PNDE can achieve 93.67% in ModelNet10, 88.74% in ModelNet40 and 94.6% in ShapNetCore13 using $M = 64$ density parameters. On the other hand, qualitative evaluations show PNDE inherits the properties of permutation invariant and rotational equivariant toward input point clouds.

Partial point cloud is a common challenge encountered in 3D acquisition on real-world object due to viewpoint occlusion and limited sensors resolution. A novel Gaussian point cloud autoencoder (GPAE) is proposed to complete a partial point cloud by sampling point cloud from inferred Gaussian components. A probabilistic sampling training strategy and weights superposition concept are designed to drastically reduce the network parameters while improving the learning efficiency. As a result, GPAE demonstrates better point cloud reconstruction in terms of Chamfer distance (CD) metric, with lowest average CD of 0.56×10^{-3} as compared to most existing models despite having up to 50 folds smaller network parameters. The GPAE can retain its efficiency toward sparse input partial point cloud by facing an average of 8% reconstruction degradation at 50% missing points, and 97% reconstruction accuracy is achieved within 5% deviation relative to ground truth.

In a challenging environment, perturbation in the input point cloud such as rotation can cause severe performance degradation in most present point cloud classification approaches. A novel part-to-whole capsule network is proposed to learn point cloud objects through parts feature reasoning using vector-based neural network, i.e. capsule network and point cloud reconstruction as learning support. In quantitative evaluation, the proposed network can achieve classification accuracy of 93.56% in ModelNet10, 88.70% in ModelNet40 and 94.71% in ShapeNetCore13. In evaluation where the input point clouds are imposed with rotation perturbation, the proposed network can achieve test accuracy of 86.67% in ModelNet10, 79.34% in ModelNet40 and 87.83% in ShapeNet13. In qualitative evaluations, the network is shown to fully reconstruct output that is equivariant to input point cloud with imposed rotation perturbation. Furthermore, the reconstructed point cloud is quantitatively evaluated using ICP algorithm and it can achieve average $\pm 2\,\mathrm{deg}$ discrepancies in rotation.

The acquisition of 3D human is highly demanded for applications such as motion capture. Conventional setups are using multiple-viewpoint for 3D human acquisition, while single-viewpoint provides flexible setup with a trade-off of partial input point clouds. Two 3D human reconstruction network models are proposed to generate 3D human model from input partial point cloud: 1) generative non-synthetic model, and 2) regressive synthetic model. The former model reconstructs complete point cloud of 3D human, while the latter reconstructs mesh 3D human. In performance evaluations, generative model can achieve an average of 32.17mm joint distance deviation against ground truth joints and 0.62×10^{-3} average Chamfer distance on reconstruction fidelity. On the other hand, the regressive model can achieve an average of 24.83mm joint distance deviation against ground truth joints and 0.84×10^{-3} average Chamfer distance on reconstruction fidelity.

Keywords— 3D Point Cloud, Parametric Mixture Model, Partial Point Cloud, 3D Reconstruction, Vector-based Neural Network, 3D Human Modeling

Publications

Parts of this thesis and concepts from it have been previously submitted in the following journal or conference papers.

Journal Papers

1. Jonathan Then Sien Phang, King Hann Lim, and Raymond Choo Wee Chiong, "Point Cloud Neural Density Estimation", submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence. (Revision) [Content as documented in Chapter 3]
2. Jonathan Then Sien Phang, and King Hann Lim, "Gaussian Point cloud Autoencoder for Partial Point cloud Reconstruction", submitted to Elsevier Journal Image and Vision Computing. (Revision) [Content as documented in Chapter 4]
3. Jonathan Then Sien Phang, King Hann Lim, and Po Ken Pang. "Skeletal Joints-based Regressive 3D Human Reconstruction from Partial Point Cloud." Journal of Information Science and Engineering, Vol. 38 Issue 5, p963-975. 13p. 2021. https://doi.org/10.1109/GECOST52368.2021.9538629 [Content as documented in Chapter 6]
4. Jonathan Then Sien Phang, King Hann Lim, and Raymond Choo Wee Chiong. "A review of three dimensional reconstruction techniques." Multimedia Tools and Applications (2021): 1-13. MTAP-D-20-01037R1. 2021. https://doi.org/10.1007/s11042-021-10605-9 [Content as documented in Chapter 2]

Conference Papers

1. Jonathan Then Sien Phang, and King Hann Lim, "Point Clouds Object Classification using Part-based Capsule Network", In 2021 Inter-

national Conference on Cyber Warfare, Security & Space Research, pp. 239-247. 2022 [Content as documented in Chapter 5]

2. Jonathan Then Sien Phang, King Hann Lim, and Po Ken Pang. "Generative Skeletal Joint-Based Autoencoder for 3D Human Point Clouds Reconstruction." In 2021 International Conference on Green Energy, Computing and Sustainable Technology (GECOST), pp. 1-6. IEEE, 2021. https://doi.org/10.1109/GECOST52368.2021.9538629 [Content as documented in Chapter 6]

Attribution Statements:
All authors provide equal contribution to the completion of the publication listed above.

Contents

List of Figures

List of Tables

List of Acronyms

3D	Three Dimensional
AE	Auto-Encoder
ANN	Artificial Neural Network
CD	Chamfer Distance
CDF	Cumulative Distribution Function
CNN	Convolutional Neural Network
DGCNN	Dynamic Graph Convolutional Neural Network
DL	Deep Learning
DR	Dynamic Routing
EM	Expectation-Maximization
EMD	Earth Mover Distance
FAUST	Fine Alignment Using Scan Texture
FPFH	Fast Point Feature Histogram
GAN	Generative Adversarial Network
GDE	Global Density Encoder
GE	Gaussian-based Encoder
GMM	Gaussian Mixture Model
GPAE	Gaussian Point Cloud Auto-Encoder

ICP	Iterative Closest Point
LDA	Local Density Auto-Encoder
LRF	Local Reference Frame
MAP	Maximum A Posteriori
MHAD	Multimodal Human Action Database
MLE	Maximum Likelihood Estimation
MLP	Multiple Layer Perceptron
MMCL	Margin Contrastive Loss
MSE	Mean Square Error
MSV	Multiple Stereo Vision
PCN	Point Cloud Completion Network
PD	Patch-based Decoder
PDF	Probability Distribution Function
PFH	Point Feature Histogram
PIFT	Perspective Invariant Feature Transform
PNDE	Point Cloud Density Estimator
PSN	Part Sampler Network
RoPS	Rotational Projection Statistics
SDF	Shape Diameter Function
SJE	Skeletal Joints Encoder
SMPL	Skinned Multi-Person Linear
STAR	Sparse Trained Articulated Human Body Regressor
STN	Spatial Transformer Network
VAE	Variational Auto-Encoder
VR	Virtual Reality

Chapter 1

Introduction

1.1 Project Overview

Three-dimensional (3D) point clouds representation learning using Deep Learning (DL) has drawn huge interest in recent years due to the rapid development in 3D acquisition technologies making 3D sensors more accessible and affordable, such as Microsoft Kinect [1] and Intel Realsense [2]. Applied DL on 2D and 3D data modality have led to many successes in challenging real world applications such as object recognition [3, 4] and segmentation [5–7], autonomous navigation and localization [8–10], semantic scene understanding [11–13], and 3D human modeling [14–17]. DL architectures on 2D data have achieved remarkable results, however 2D data comes with limitations in perceiving depth which is critical in depth estimation and localization. Therefore, depth sensing devices emerge to provide rich geometric and depth information in the form of 3D data. The depth sensing devices can be broadly categorized into three types [18, 19], i.e. Time-of-Flight (ToF), structured-light, and stereo-vision sensors. ToF-based sensors, for eg. Light Detection and Ranging (LiDAR) measures the time of emitted light from illuminator that reflects back from an object to calculate a depth map. Structured-light sensors project an infrared light pattern onto an object to estimate the disparity of perspective pattern distortion for depth map calculation. Stereo vision uses multiple passive sensors to capture the reflection of natural light to compute the disparity between two viewpoints in perceiving the depth information. These range sensors commonly store the data in the format of raw point clouds [19, 20], RGB image coupled with depth map (RGB-D) [21] or an intermediate derived 3D data representations [22] such as voxels and meshes.

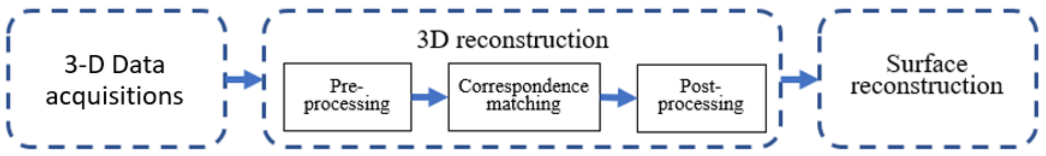


Figure 1.1: Generalized 3D modeling pipeline. [23]

A generalized pipeline of conventional 3D modeling consists of 3D data acquisition, 3D reconstruction and surface reconstruction as illustrated in Figure 1.1. Recent research and implementation have shifted the focus in potential of using multiple stereo vision acquisitions (MSVs) due to the richness information of input data in 360-view. MSVs contain diverse 3D data output format, i.e. color images associated with depth map (RGB-D) and point clouds. The RGB-D format typically contains four channels of images, i.e. three channel RGB images and one channel depth map, while point clouds format are typically presented in three dimensions (X, Y, Z) in Euclidean space. In general, advanced 3D reconstruction techniques can be divided into three types, i.e. statistical models, discriminative learning models and generative learning models. Statistical models [24–27] use mathematical tools to express the geometric information in a 3D handcrafted feature descriptor. This descriptor is designed solely to describe specific geometry using underlying statistic characteristic of input data. A corresponding matching is performed between multiple set of features to search for the maximum likelihood region and compute the best resultant matching affine transformation. However, the handcrafted feature descriptors are often restricted by its capacity of expressing and generalizing novel data that are not tailored to its input pattern. Therefore, a more generalized technique should be developed to adapt to the data pattern using deep learning approach.

Recent success in learning-based approach using scalar-based neural network [4, 10, 28–32] has enabled the formulation of versatile data-driven descriptor to establish correspondence in multi-dimensional data. Learning-based 3D descriptor [4, 28–30] focuses on establishing global descriptors in various representation of 3D data. They are specialized in global level features learning and hence are ideal to be implemented in object classification and segmentation task. On the other hand, 3D local descriptor [10, 31, 32] is designed to inherit advantages of data-driven in learning-based approach and properties of learning local level feature to establish correspondence in non-rigid deformed data. However, conventional 3D reconstruction techniques are constantly facing issues in deformations of the input resulting an addition degree of difficulty in the data processing. In light of generative learning models [33–36], learning-based 3D descriptor approaches are implemented to directly process raw input 3D data and reconstruct output in generative manner using autoencoder architecture [33]. Derived from unsupervised clustering based on the probabilistic distribution, i. e. Gaussian distribution, the autoencoder architecture is modeled to explore and discover the underlying latent distribution of input data. Nonetheless, existing generative learning models [34–38] adopt scalar-based neural network to perform features learning to produce data representation.

Due to scalars inability to efficiently represent spatial and pose features without addition parameters, they are often required to estimate the canonical pose of input data prior to processing [4, 39]. Inspired by the capsules network architecture [40, 41], vector representation can be adopted in 3D local descriptor to efficiently encode spatial and pose features. The vector-based neural network introduces the concept of equivariance by retaining the pose information to the encapsulated local descriptor that encodes input features. This property can greatly improve a generative reconstruction, particularly in novel viewpoint of the input and circumvents the use of estimated canonical pose [42]. Thus, the motivation of this research work is to design novel generative learning model that can learn raw 3D point cloud representation, point cloud reconstruction and for real-world 3D modeling application. In the following sections, the problem statements, aim and objectives are laid out in detail to highlight the challenges and resolution proposed in this research work.

1.2 Problem Statement

The modeling of 3D object plays a crucial role in enabling spatial and depth visualization in important tasks such as autonomous robotic navigation, object recognition, Virtual Reality (VR) and human bio-mechanics understanding. Due to the advancement of depth sensing technology, 3D data has become a preferred data modality over 2D data in presenting an object because of the capability to retain depth information. However, 3D data representation learning and reconstruction techniques have encountered multiple challenges due to variations in the data representation, characteristics and properties. Consequently, the variations can greatly affect the procedure in analyzing and processing the 3D data to efficiently reconstruct a 3D model. In brief, the problems could be structured into three aspects as follows:

1. Number of acquisition equipment is increased as a counter measure to provide coverage for occluded viewpoint of an object instance [43, 44]. With increased number of equipment in multiple-view acquisition, computation cost is extensively increased due to larger stream of incoming data and required computation power. Multiple-view acquisition can be branched into marker-based and markerless-based approach. Pioneering marker-based 3D modeling system such as VICON [45] and Qualisys [46] require active or passive markers to be attached on moving object. A precise equipment setup and large operating environment are also required to do 3D data acquisition. On the other hand, markerless-based approach applies computer vision techniques to acquire object's depth information using commodity range-sensing sensors such as stereo vision. Moreover, unequal distance placement of equipment can cause non-rigid deformation in the acquired 3D data in each viewpoint. Non-rigid deformations are associated with spatial and geometric deformation, i.e. scaling and stretching. In addition, non-stationary object, particularly in large motion can further increase the complexity of non-rigid deformation. Due to recent research advancement in deep learning architectures, obtaining complete 3D data information from single viewpoint has drawn attentions in many 3D modeling applications [47],
2. Widely implemented 3D reconstruction techniques can be generalized into two categories: 1) Statistical model, and 2) Discriminative learning model. In statistical model, the technique mainly depends on statistical analysis on the input data using mathematics derivation to establish correspondences between viewpoints. However, statistical model is poor in novel data generalization due to lack of learning capability. For instance, statistical models lack capability in handling higher complexity geometric structure. The latter, discriminative model incorporated Deep Learning (DL) architecture in performing data analysis and learning to establish correspondences of input data. In comparison, discriminative model outperformed statistical model in establishing correspondences with higher capability in novel data generalization. However, the discriminative learning model is performing at a less ideal correspondence accuracy within 60-67% in establishing correspondences for 3D reconstruction. Moreover, discriminative model remains as a conventional 3D reconstruction technique where challenges such as rigid and non-rigid deformation imposed on input data can greatly affect its performance. On the other hand, recent research trend in generative learning models

have shown its capability to generate realistic output and great generalization to high degree of shapes varieties. Hence, efforts are focused on generative learning models in 3D reconstruction due to its performance, feasibility and applicability,

3. Discriminative learning model is a stepping stone in DL-based 3D reconstruction by leveraging the learning capability of DL to process and learn 3D data. Due to many challenges faced in conventional 3D reconstruction techniques, recent research efforts have been invested into generative techniques for 3D reconstruction. Thanks to the advancement in technology of depth sensing, 3D datasets are widely available that are covering many category of objects, scene and environments. Benefiting from the abundance of 3D data, DL techniques can leverage the large variations in 3D datasets to achieve better generalization in generative reconstruction. However, most current generative learning models in point cloud reconstruction are using scalar-based neural networks [34, 35] in generating scalar features as latent representation. As the result, scalar features cannot substantially represent spatial and geometric properties of 3D data due to lack of pose information and lesser immunity to outliers and noises [48, 49]. In addition, current generative learning models [35, 36, 38] are depending on large dimension in latent representation which can greatly impact on the network complexity and parameter .

1.3 Aim and Objective

The aim of this research is to develop a new 3D point cloud representation learning and reconstruction technique to improve the current 3D modeling using raw input point cloud acquired from single viewpoint depth sensor. The objectives of this research can be divided into three sub-objectives as follows:

1. To design a novel 3D point cloud representation learning and reconstruction technique driven by learning of local descriptor, i.e. Gaussian components and generative sampling using DL architecture derived from point cloud density distribution,
2. To design a part-to-whole DL architecture by adopting a novel vector-based neural network pipeline to encapsulate local descriptors into global latent features (global descriptor) for 3D point cloud reconstruction,
3. To verify the proposed 3D point cloud representation learning technique and generative learning model in real-world application of 3D modeling using input

data acquired from single viewpoint depth sensor.

1.4 Significance and Contributions

Representation of an object in 3D is crucial for aggregating its shape and geometrical information in multiple viewpoints. In this research, several novel network architectures are proposed for 3D data representation learning and reconstruction for the applications in 3D modeling. The research works explore the combination use of density estimation in point cloud, generative sampling for point cloud reconstruction, part-to-whole learning DL architecture and application in 3D modeling using real-world data. The significance and contributions of this research can be summarised as follows,

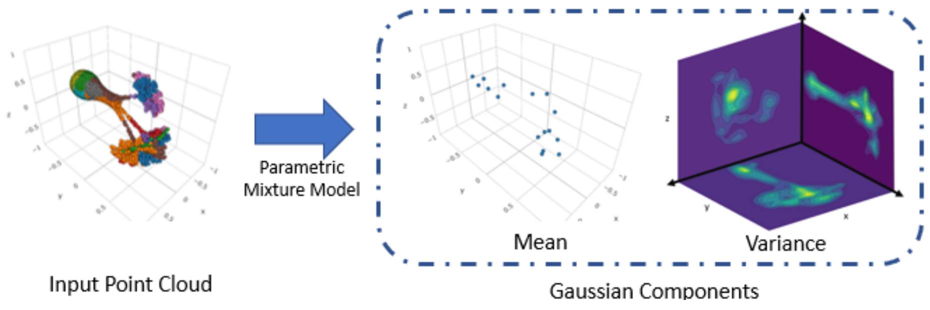


Figure 1.2: Point cloud representation learning using parametric mixture model.

1. The density distribution of point cloud is crucial in representing its underlying structure in a reduced and concise manner. Learning representation of point clouds using neural networks poses challenges due to the sparsity and its unordered structure. Moreover, conventional point cloud models with single latent feature representation [34, 35, 37] are not sufficient to fully capture the complex geometry of a point cloud. A key contribution to this research work is a proposed parametric mixture model based on a neural density estimation to produce multiple local representations of a point cloud, where each representation is a density parameter as shown in Figure 1.2. In particular, Gaussian mixture model is the key element in estimating the local feature distribution on a point cloud using Gaussian components, i.e. mean and variance. Conventionally, iterative computations are required to estimate density parameters in a Gaussian mixture model. By using neural network to produce latent inference, the iterative term can be eliminated and resolved by network training iterations. The proposed network

named Point cloud Neural Density Estimator comprises two-stage network modules, (a) Global Density Encoder encodes the latent global density of input point cloud, (b) Local Density Autoencoder first infers latent local densities from latent global density through latent inferencing, subsequently a projection network is used to project ambient density parameters. The network is trained in the unsupervised manner by directly maximizing the average log-likelihood function derived from Gaussian mixture model. The significance and contribution of this research work is journaled in Chapter 3.

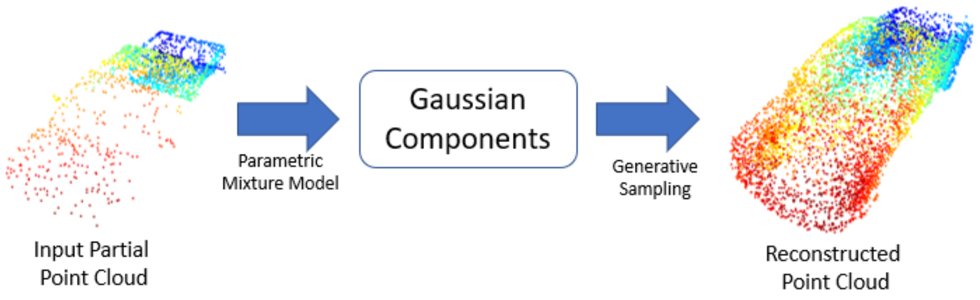


Figure 1.3: Partial point cloud reconstruction from Gaussian components using generative sampling.

2. Partial point cloud is a common challenge encountered in 3D data acquisition on real-world object due to viewpoint occlusion and limited sensors resolution. Complex configuration in multiple viewpoints acquisition may not be practical in achieving real-time operation. Therefore, partial point cloud reconstruction techniques using single viewpoint are recently drawn into focus by leveraging the learning capability of deep learning neural networks. A key contribution is realized in a novel Gaussian point cloud autoencoder to reconstruct a complete point cloud from partial point cloud inspired by generative sampling process using estimated Gaussian components of a ground truth point cloud as illustrated in Figure 1.3. As opposed to existing work that utilize shared single latent feature for reconstruction and residual network for point cloud completion, this work uses ambient Gaussian components as local feature representation for complete point cloud reconstruction. Moreover, a probabilistic sampling training strategy and weights superposition concept are proposed, drastically reducing the network parameters and improving the learning efficiency. As a result, the proposed model demonstrated better point cloud reconstruction as compared to most existing models despite having a smaller number of network parameters. In addition, the

network has an advantage of providing observable insight of local features i.e., mean and variance, that can be reviewed in ambient space. The significance and contribution of this research work is journaled in Chapter 4.

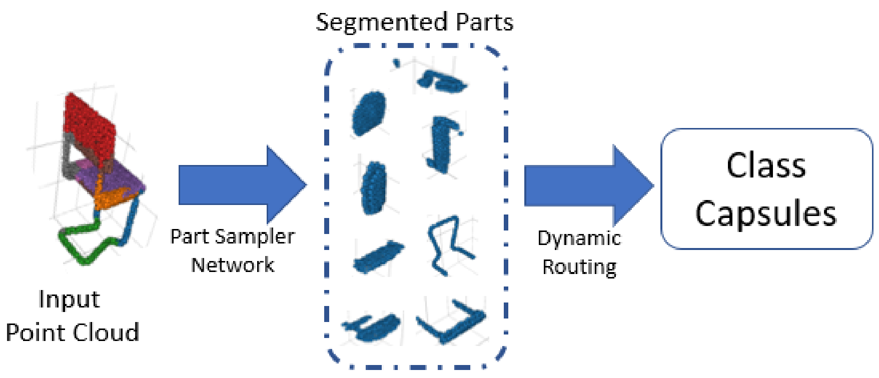


Figure 1.4: Part-to-whole learning through Dynamic Routing on segmented local point cloud parts.

3. Point cloud object classification task has become trending due to crucial applications using depth sensors in fields such as autonomous robotic navigation and semantic environment learning. With recent advancement in DL architectures, more research efforts have been invested in achieving efficient and feasible depth information processing. However, most DL architectures are relying on scalar-based neural network to encode features of input point cloud. Due to lack of ability to encode pose information in scalar-based neural network, the network performance can be tremendously impacted when input point cloud is imposed with a novel transformation, i.e. rotation. A key contribution in this research work is realized in a novel part-to-whole capsule network to learn point cloud objects through parts feature reasoning as illustrated in Figure 1.4. To efficiently obtain part features (local descriptor) from input point cloud, local parts are segmented using proposed Part Sampler Network (PSN) and encoded into part capsules. Through dynamic routing algorithm, the network learns to determine the existence of an object class capsule (global descriptor) via parts voting. Benefited from capsule architecture, the features are encoded in vector form, where the magnitude of vector represents the activation of feature and the direction represents the pose information. Therefore, the proposed network inherited the equivariant property in point cloud representation learning and reconstruction. Lastly, a reconstruction loss is incorporated in addition to existing objective func-

tion to further enhance the learnt object class feature by adopting the proposed Gaussian point cloud reconstruction network. The significance and contribution of this research work is journaled in Chapter 5.

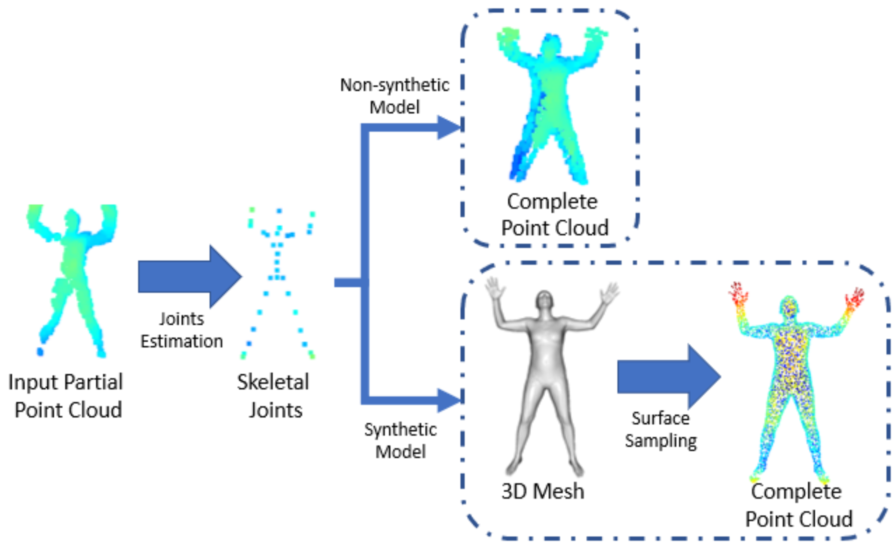


Figure 1.5: 3D human reconstruction models using input partial point cloud of scanned human.

4. Partial viewpoint occlusion is a common issue encountered in acquiring 3D human model using single viewpoint acquisition. The use of generative learning model is recently drawn into point clouds reconstruction to infer complete point cloud due to its generative learning property. A key contribution is realized in two proposed 3D human reconstruction techniques using input partial point cloud acquired from single viewpoint depth sensor as illustrated in Figure 1.5. The proposed 3D human reconstruction techniques can be categorised into two reconstruction models: (1) Generative skeletal joint-based Autoencoder for 3D human point cloud reconstruction (Non-synthetic model), (2) Skeletal joints-based regressive synthetic 3D human reconstruction (Synthetic model). In non-synthetic model, the proposed autoencoder architecture is comprised of a joints encoder and a patch decoder. The joints encoder first learns the latent representation of input partial point cloud using Gaussian maximum likelihood and infers a set of Gaussian components consisting of the localized skeletal keypoints and variances of ground truth human. Subsequently, the inferred Gaussian components are viewed as local human part keypoints for local patch sampling using a patch

decoder. In synthetic model, the proposed network infers skeletal joints and variance for synthetic 3D human reconstruction. Similar to non-synthetic model, localized skeletal joints and variances of ground truth human are estimated using a joints encoder. Subsequently, the skeletal joints are fed to a regressive human model to reconstruct a synthetic human model to obtain a complete point cloud. Lastly, a two-mode training strategy is proposed to enhance the learning of the proposed method by employing synthetic training in prior and non-synthetic fine-tuning. The significance and contribution of this research work is journaled in Chapter 6.

1.5 Thesis Overview

This thesis provides a detailed study and development on 3D point cloud learning and reconstruction, and evaluation using various benchmark syntethic and non-synthetic datasets. It is outlined into six chapters as follows:

Chapter 2: ***Literature Review on 3D Point Cloud Representation Learning and Reconstruction***

This chapter covers the latest review of 3D point cloud reconstruction techniques in 3D modeling, i.e. statistical model, discriminative learning model and generative learning model. A detailed comparison on the three types of 3D point cloud reconstruction techniques are reviewed in term of input data structure, correspondence accuracy, precision and recall using four benchmark datasets, i.e. ModelNet10, ModelNet40, ICL-NUIM, and Semantic3D. The advantages and disadvantages of 3D point cloud reconstruction techniques are highlighted for implementation guideline and future improvements.

Chapter 3: ***Point Cloud Neural Density Estimation***

Learning and analyzing point cloud using DL neural network is challenging due to the sparsity and un-ordered nature of the data modality. This chapter proposes a parametric mixture model based on a neural density estimation to produce multiple local representations of a point cloud, where each representation is a density parameter. In particular, the proposed network is inspired by Gaussian mixture model in extracting local feature distribution of the sparse and un-ordered point cloud using Gaussian components. Due to the property of local maximizer of Gaussian distribution, the proposed network inherited the property of equivariant, which is a desired to retain input transformation through out the network propagation. The performance of proposed network is evaluated on the benchmark datasets: Modelnet10, Modelnet40, and ShapeNetCore13.

Chapter 4: ***Partial Point Cloud Completion using Patch-based Autoencoder***

Partial point cloud is a common challenge encountered in acquiring 3D data on single viewpoint depth sensor acquisition. This chapter covers the development of partial point cloud completion technique inspired by generative sampling process using point cloud representation derived from Gaussian components. A patch-based decoder is proposed to reconstruct output point cloud by taking input of Gaussian components. Probabilistic sampling training strategy and weights superposition concept is proposed to increase the learning capability and reduce the network parameters. The performance of proposed network is evaluated on ShapNetCore synthetic dataset and KITTI non-synthetic dataset.

Chapter 5: ***Part-to-Whole Learning Capsule Network on Point Cloud Classification and Reconstruction***

Learning equivariant representation of point cloud is challenging due to limitation in scalar-based neural networks. This chapter demonstrates a vector-based neural network for equivariant point cloud representation learning and reconstruction by adopting capsule network. The proposed network incorporated point cloud density estimator for part segmentation and generative sampling decoder for reconstruction. In addition to conventional capsule network objective function, the proposed network is trained end-to-end using joint objective function. Performance of proposed network is evaluated and compared to existing scalar-based network on benchmark datasets i.e. ModelNet10, ModelNet40, and ShapNetCore13. The proposed network is also evaluated on classification and reconstruction of perturbed input point cloud.

Chapter 6: ***Skeletal Joints-based 3D Human Reconstruction from Partial Point Cloud***

3D human acquisition is a challenging task, particularly due to single viewpoint setup and dynamic movement of the human during scanning. This chapter covers the implementation of generative learning model proposed in previous chapters to reconstruct 3D human model by taking input of partial point cloud. Two 3D human reconstruction models are proposed in this chapter, where the models cover both non-synthetic and synthetic output of 3D human model. Both 3D human reconstruction models first estimate the skeletal joints of human. To generate non-synthetic output of 3D human, the estimated skeletal joints and its variances are fed to a generative decoder to reconstruct a complete point cloud of human. On the other hand, synthetic output of 3D human is generated by input the skeletal joints into a regressive 3D human generator to regress a 3D human model. Uniform complete point cloud of human can be obtained by surface sampling the 3D human model. Further, a two-mode training strategy is proposed to enhance the network learning by training the network using synthetic data in prior and fine tune the network using non-synthetic data. The performance of the proposed 3D reconstruction models are evaluated on a non-synthetic dataset.

Chapter 7: ***Conclusion and Future Work***

Novel 3D data representation learning and reconstruction techniques are proposed to improve the efficiency of point cloud representation learning as well as in reconstruction. A summary of every chapter is drawn in this chapter to highlight the contributions and significance of research findings. The advantages and limitation of the proposed 3D data representation learning and reconstruction techniques are discussed in the chapter for future development.

Chapter 2

Literature Review on 3D Point Cloud Representation Learning and Reconstruction

2.1 Introduction

Three-dimensional (3D) point cloud representation learning and reconstruction using Deep Learning (DL) architecture has become an active trend in the research field of computer vision. The main advantage of using DL architectures to analyse information is their capability to learn supervised or unsupervised with provided information source. A DL architecture, i.e. neural network, can provide a framework where a representation of a set of complex information can be generated by training the network. By having more advantages in representing object in 3D space compared to conventional 2D data, recent researches are shifting into analyzing 3D representation information of object using DL. With variety of 3D representations, such as voxels, RGB-D, projections, and point cloud, they pose many complex challenges not encountered in conventional 2D data. Among the representations, point cloud gained the most interest by researchers due to its simplistic representation in three dimension space and contain rich geometric detail. However, point cloud is coupled to several challenges such as sparsity and points permutation [4], noises and outliers [50], and partial data [35, 36]. Despite the challenges, point cloud processing using DL architectures has recently become an active research area in learning rich representation from point cloud mainly due to advancement and accessibility to affordable depth sensors. Moreover, completion of point cloud is also a great interest in recent research trend to concisely express an object in 3D modeling and as part of a vital step in 3D reconstruction. Pioneering works commonly employ one of several, such as latent features [4, 51, 52], density model [53, 54], and variational density model [55] to encode a point cloud to perform specific tasks, such as object classification and reconstruction. Unsupervised learning in the point cloud processing models [51, 53, 55–57] are also commonly employed to learn the distribution of the point cloud corresponding to the variation in the data.

A basic process of 3D modeling comprises of depth map acquisition, 3D reconstruction and surface reconstruction [23]. 3D reconstruction techniques are the core computing process in 3D modeling to generate a complete 3D object model. The process typically generates representations of point cloud in prior and subsequently post process the point cloud to complete a 3D object model. The process models can be broadly divided into three categories, i.e. statistical models, discriminative learning models and generative learning models. Statistical models [26, 27, 58] use mathematical derivations to express the geometric information in a 3D handcrafted feature descriptor. This descriptor is designed to describe specific geometry using underlying statistic characteristic of input 3D data. A corresponding matching is performed between multiple set of features to search for the maximum likelihood region and compute the best resultant matching affine transformation. However, the handcrafted feature descriptors are often restricted by its capacity of expressing and generalizing novel data that are not tailored to its input pattern. Therefore, a more generalized technique should be developed to adapt to the variety of data pattern using DL approach.

Established DL approaches such as Convolutional Neural Networks (CNN) have shown promising results over 2D computer vision tasks, such as object segmentation [59, 60], object tracking [61, 62] and object classification [63–65]. Due to the limitation of CNNs [66] in modeling the sparsity and viewpoint variation of 3D data, DL models have started gaining popularity in the 3D domain attempting to better exploit the rich information in raw 3D data for 3D descriptors, while resolving its challenging properties [4, 28, 30, 67]. Several popular DL based 3D descriptors [4, 29, 30] apply supervised learning to learn shape information from predefined dataset and they are commonly known as discriminative learning models. Early development of global 3D feature learning in discriminative learning models are mainly focused on object classification [28, 66, 68], while others are focused on segmentation [4, 30]. These DL models are also known as global descriptor, and their focal objective is mainly derived from extracting semantic feature of 3D model at global level. Naturally, global descriptors fall short in establishing local correspondences that are essential in task of correspondence matching during 3D reconstruction. Due to the inability of global descriptor to efficiently learn local 3D features, DL based local 3D descriptor models are formulated to overcome the difficulties of global descriptor learning [7, 31].

Generative learning models [33] use unsupervised method based on the probabilistic distribution, i.e. Gaussian distribution, as the principled way to model machine learning and machine perception problems. In addition of Bayesian statistics to Gaussian distribution, it provided a rich and flexible language for specifying the prior knowledge of novel inputs and subsequently refining it with data and evidence [69, 70]. Conversely, discriminative learning models adjust a possibly non-distributional model to data optimizing for a specific task, such as classification or prediction. This typically leads to superior performance yet compromises the flexibility of generative modeling. The detail of 3D reconstruction techniques are reviewed in-depth to cover the commons and differences between generative learning models and non-generative approaches such as statistical models and discriminative learning models. Throughout the literature reviews of 3D point cloud representation learning and reconstruction techniques, the advantages and drawbacks of each model are discussed and followed by a summary of the future trends and development in the end of this chapter.

2.2 Fundamentals of 3D Data Representations

Depth perception by 3D data acquisitions using depth sensors can produce variety of representations with unique structure and properties. The 3D data representations can be largely categorized into two groups [22]: (i) Euclidean-structured Representation, (ii) Non-Euclidean-structured Representation. Euclidean-structured representation is typically presented in fixed grid structure such as RGB-D and volumetric data, where the data are organized by global parametization with known common coordinate system. On the other hand, non-Euclidean structured representation typically lacks of grid structure and is commonly presented in sparse 3D space and each data point is un-ordered (permutation). In the following, the two representations are elaborated in brief on their advantages and disadvantages, and interest of the representation in recent research trend.

2.2.1 Euclidean Structured Representation

Due to advancement and easy accessibility to depth sensors such as Microsoft's Kinect [1], depth information acquisition is popularized by coupling RGB image with depth map (RGB-D) [21] to obtain a 2.5D data representation, where the 0.5D represents depth map. Intuitively, RGB-D is an effective representation of a 3D object, where the RGB data retain the visual information of the scanned object, and the depth map represents the depth information in each pixel. One the other hand, a 3D object can be decomposed into multiple 3D volumetric representation (voxel) with blocks in 3D grid [29]. A voxel representation essentially describes the distribution of a 3D object in 3D space by occupying the voxel grid, thereby obtaining a voxel 3D occupancy grid. Moreover, the occupancy grid can effectively encode the viewpoint information of the scanned 3D object [22].

However, each grid in 3D volumetric representation contains sparse information while a large number of voxels inside are not useful in describing the surface feature of 3D shapes. Therefore, 3D volumetric representation is expensive and inefficient as the computation and memory are growing at cubic level, especially with increasing resolution. Similarly, 3D convolution also consumes a lot of computing power when extracting features and sampling, which also results in low and limited resolution for most 3D voxel reconstruction tasks. In addition, converting point cloud into 3D volumes introduces a quantization effect that discards some details of the data [71, 72] and is not suitable for representing fine-grained information.

2.2.2 Non-Euclidean Structured Representation

Two main types of non-Euclidean structured representation are 3D mesh and point cloud. Essentially, 3D mesh is a collection of vertices and faces that construct a 3D shape. The vertices are a set of points associated in a list of ordered index that that is similar to a grid structure in 3D space. The ordered vertices also describe the connectivity of each point that ultimately constructs the surface of a 3D shape. Due to the nature of connectivity of vertices in 3D mesh, it has close relationship to a graph representation, where the connectivity can be viewed as the edge of each vertex. However, 3D mesh is mostly only available as a post-processed 3D representation such as synthetic 3D object model.

In raw 3D representation, point cloud can represent 3D shapes in a more efficient manner. As point cloud is sampled from surfaces of objects, it can capture the details of surfaces of objects. A set of 3D point cloud represents a geometric shape, where the surface is typically composed of coordinates in 3D space and is defined in format of (x, y, z) [4]. Thus, a 3D representation of an object or scene is viewed as a $3 \times N$ matrix, where N is the number of points. 3D point cloud as an input modality poses a unique set of challenges (e.g. permutation, rotation and data artifacts) when devising a network architecture [4]. Noises, outliers and partial data distribution are commonly associated in data artifacts and the occurrence of data artifacts is unavoidable in the real world applications[50]. In addition, point cloud can be processed to obtain intermediate representation such as voxels or meshes to offer benefits in certain task specific procedures [23].

2.3 Point Cloud Representation and Reconstruction

Let a set of 3D point cloud defined as $\mathcal{P} = \{p_i \in \mathbb{R}^3\}_{i=1}^N$, where $\mathcal{P}$ represents a 3D object. 3D reconstruction can be formulated as follows: (i) A 3D object P can be reconstructed or generated by a technique $\mathcal{F}$. (ii) A ground truth 3D object $G = \{g_i \in \mathbb{R}^3\}_{i=1}^N$ exists to quantify performance metrics of $\mathcal{F}$ and it can be decomposed into non-generative and generative models. Generally, $\mathcal{F}$ follows a generic pipeline, i.e. pre-processing, correspondence matching and post-processing [23]. Pre-processing and post-processing are essential in non-generative approaches to overcome aforementioned challenges before processing the data. On the other hand, correspondence matching is the integral part of non-generative models, which is responsible in establishing correspondence keypoints by finding similarities in 3D data point as shown in Fig. 2.1. Typically, it requires more than one viewpoint to construct a 3D model in non-generative models. Therefore, performance of correspondence matching is more weighted in non-generative models to enable high quality 3D reconstruction [10].

In non-generative models such as statistical models and discriminative learning models, the correspondence matching can be defined as $R(\cdot)$ function below:

$$R(c_1, c_2) = |d_{dis}(f, f')| \leq \tau, \tag{2.1}$$

where c_1 and c_2 are two local patches containing multiple points of interest. Hence, the correspondences are computed using dissimilarity metric d_{dis} between f and f' feature points generated by a feature descriptor function $\mathcal{X} : c_1 \to f$ and $c_2 \to f'$ respectively. τ is a threshold of correspondence probability value. Lower value of dissimilarity metric

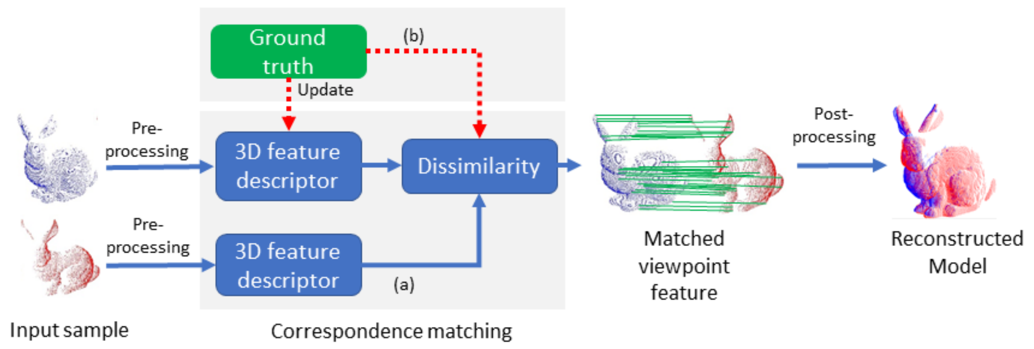


Figure 2.1: The basic architecture of non-generative 3D reconstruction models using multiple viewpoint partial point cloud with the indicating flow path: (a) statistical models use mathematical equation to derive 3D feature descriptor in a forward manner, (b) Discriminative learning models apply ground truth data as a supervised model to obtain trainable feature descriptor in an iterative backpropagation manner.

than τ indicates that c_1 and c_2 are likely to be the correspondence on the same segment of a 3D object and this is also reciprocal to maximum likelihood matching when similarity metric is used. In statistical models, $\mathcal{X}$ is a primitive approach that relies on handcrafted function derived from mathematical expression as shown in Fig. 2.1(a). On the other hand, in discriminative learning models, $\mathcal{X}$ is a supervised data-driven DL neural network governed by the dissimilarity metric as an objective function to generate the corresponding features as shown in Fig. 2.1(b).

However, generative models do not rely on $\mathcal{X}$ for correspondence matching. Intuitively, generative models adopt unsupervised data-driven DL networks to learn input feature distribution involving an encoder network $\mathcal{E}$ and generate an output that has close approximation to input data involving a decoder network $\mathcal{D}$[73, 74], such that

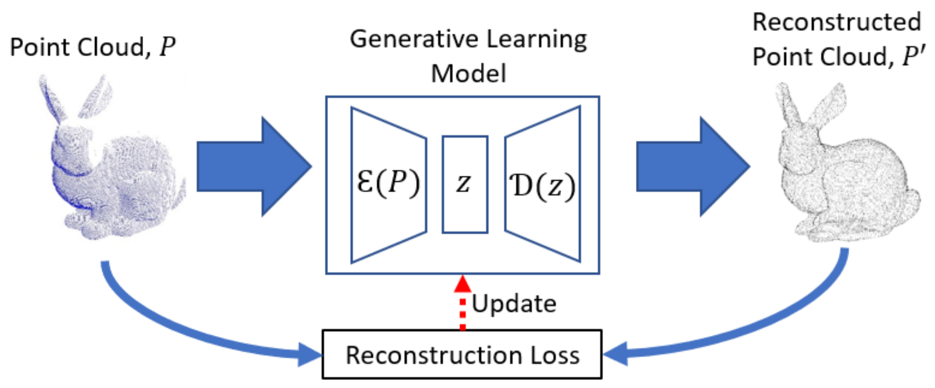


Figure 2.2: The flow of generative learning models governed by a reconstruction loss to generate a full 3D model from partial point cloud in single viewpoint.

$$z \sim \mathcal{E}(P) \tag{2.2}$$

$$P' \sim \mathcal{D}(z) \tag{2.3}$$

where $\mathcal{E}(\cdot)$ is a network that infers latent distribution z of input data P. $\mathcal{D}(\cdot)$ is a network that samples output data P' from the latent z. A reconstruction loss is used to govern the unsupervised training mechanism of the generative learning models in order to obtain a generic 3D modeling as shown in Fig. 2.2.

2.3.1 Statistical Models

Statistical models use handcrafted mathematical derivation to establish image feature descriptor from the underlying statistical properties, such as image edges or corners [75]. Several early statistical models [76–78] develop 2D feature descriptors for image classification and pattern recognition. At the present, 2D feature descriptors are ill-performing in 3D domain due to data sparsity and rotational variant [4] which do not exist in 2D data. Therefore, 3D feature descriptors are devised to model the spatial and geometric structure of 3D data. However, 3D feature descriptors [58, 79, 80] are operated at global level, hence they are not ideal for higher complexity geometric structure representation at local level. Subsequently, local level 3D descriptors [25–27, 81–83] are formulated to overcome the disadvantages of global 3D feature descriptors, while improving the performance of correspondence matching.

In Point Feature Histogram (PFH) [81], a histogram-based 3D descriptor is proposed to model the geometry structure of 3D data at local region level. In an incremental development of PFH, Fast Point Feature Histogram (FPFH) [25] is optimized from PFH to effectively reduce the computational footprint, while retaining the discriminative properties. Subsequently in SHOT [26], a 3D local descriptor featuring a hybrid structure of signature and histogram is proposed, which incorporates geometric signature coupled with histogram features. As the result, SHOT is able to achieve a more descriptive feature descriptor then FPFH. Moreover, a shape diameter function (SDF) is introduced by Shapira et al. [82] to represent contextual signature of local shape volumetric information. Meanwhile, a probabilistic descriptor is proposed [84] to jointly interpreting surface correspondences, segments, and shape deformation to work in correlation to achieving an efficient descriptor.

On the other hand, Local Reference Frame (LRF) descriptor and Rotational Projection Statistics (RoPS) descriptor is proposed by Guo et al.[83] to describe a viewpoint-invariant local geometric and represents distribution of the rotationally projected adjacent feature points. Both LRF and RoPS descriptors are to overcome viewpoint and rotation invariant issue respectively when analyzing arbitrary region of 3D data. In a more recent work, Perspective Invariant Feature Transform (PIFT) [27] is proposed to incorporate color and depth information prior to feature extraction. Using color and depth information, the PIFT is able to filter uncorrelated correspondence keypoints that are unstable in correspondence matching. As a summary, statistical models are computational efficient due to their discrete implementation. However, they are highly dependent on specific task to derive 3D feature descriptors from the input data. Besides, statistical models lack of learning process which will enable the model to adapt to novel data. Therefore, statistical models are unable to generalize in new data modalities or scale accordingly.

2.3.2 Discriminative Learning Models

The development in DL approaches has been progressing tremendously with remarkable performances in various applications due to the advancement on optimization techniques and DL architectures [85]. One of the DL approaches in discriminative learning models is in their supervised data-driven mechanism to learn features descriptor from ground truth data. They can achieve better generalization towards novel data and are more efficient as compared to statistical models. By using the advantages of parallel computation, these models are then specifically tailored to analyse and process 3D data. Early works in discriminative learning models [4, 28–30, 66, 67, 86] showed that the feasibility of DL approaches can be applied in 3D data, such as 3D point cloud, meshes and voxel grids. A general implementation of a discriminative learning model is using specialized neural network i.e. CNNs to learn an overall feature representation of 3D data and in combination with a fully connected neural network for classification task. Although most of the early works put focus on performance in classification task, they mainly learn global level representation, which is less ideal for local correspondence matching in 3D reconstruction.

Processing 3D data with grid structure such as voxel representation is straight forward because 3D convolutional neural networks (3D CNNs) can be directly implemented to analyze the data. Early 3D reconstructions [66, 87–90] apply 3D CNNs build upon the volumetric representation of 3D point cloud. Giridhar et al. [87] learnt a joint embedding of 3D voxel shapes and their corresponding 2D images. Choy et al. [91] trained a recurrent neural network to encode information from more than one input views. Besides that, Xie et al. [38] introduced 3D grids as intermediate representations to regularize un-ordered point cloud and proposed a novel Gridding Residual Network (GRNet) for point cloud completion. They devised two novel differentiable layers, named Gridding and Gridding Reverse, to convert between point cloud and 3D grids without losing structural information. They also presented the differentiable cubic feature sampling layer to extract features of neighboring points, which preserves context information. In addition, a new loss function, namely Gridding Loss was designed to calculate the L1 distance between the 3D grids of the predicted and ground truth point cloud, which is helpful to recover details. But voxel formats are computationally heavy and information sparse, which lead to research on the octree data structure [30, 92, 93] for representing 3D data. Despite the simplicity in representing 3D data using octree data structure, they contain finer details of 3D objects with less computation compared to voxel. However, both voxel and octree representations fall short in their capability in preserving geometry of 3D objects such as surface smoothness and intrinsic shape properties [22].

In more recent researches, efforts have been invested in direct process raw 3D point cloud despite the challenges using DL. In Kd-Network[67], the approach represents a geometric structure of unstructured 3D point cloud by modeling neighbourhood points as hierarchical graph. Additionally, Khoury et al. [94] constructs spatial semantic histograms on adjacent points of each unstructured 3D data point. Using constructed spatial semantic histogram as local 3D feature, it is used as training data for a neural network to learn discriminative local feature representation. However, they mainly rely on statically pre-processed 3D point cloud as the input source for their DL architecture. In contrary, PointNet [4] and PointNet++ [7] are the pioneering works that proposed shared MLP-based networks on ingest unstructured 3D point cloud analysis and tasks such as 3D object models classification and segmentation. Due to its property in permutation invariant and ability to directly ingest raw point cloud, Pointnet is regularly implemented as point cloud encoder for latent representation learning [7, 32, 35]. On the other hand, a directed graph with vertices and edges can aggregate local features in a point cloud regardless of the sparsity and points permutation, where vertices are the points and edges are the distance feature of neighbourhood points. In graph-based models [95, 96], a directed graph is constructed from the input point cloud and a neural network is used to project the directed graph into latent features. In Dynamic Graph CNN (DGCNN) [52], the proposed work constructs hierarchical directed graph, of local neighborhood points in a point cloud and applies convolution-like operations on the graph to encode richer latent features.

More recently, Siamese network came to the attention of researchers as it is architecturally designed to maximize the discrimination learning, which can assist the network to learn fast and efficiently [60, 97, 98]. A Siamese network architecture is constructed using two identical networks containing identical weights in both networks. Each network in a Siamese network producing a comparable output vector that will be computed for dissimilarities. Simple distance metrics such as L_1 distance can be implemented to compute dissimilarities [97]. Forthwith, Siamese network is widely adopted as the discrimative learning model in [10, 31, 32, 99]. In 3DMatch [10], Siamese 3D CNN is implemented to infer discriminative local features from local voxel grids. L_1 distance is used to compute dissimilarities between patches to find correspondences between patches. [31] on the other hand learns to generate discriminative features by using projection of 3D data and a contrastive loss as objective function. Furthermore, DeepPoint3D [32] uses PointNet as the base architecture to directly ingest 3D point cloud. A novel Multi Margin Contrastive Loss (MMCL) is proposed in DeepPoint3D network to leverage non-convergence and fast-convergence data in a joint objective function. As the result, DeepPoint3D network could learn better generalization in inferring discriminative local feature. On the other hand, LRF-net [99] utilizes the discriminative advantage of a Siamese network on LRF as the training data. As discussed in statistical models, a LRF is a local geometric descriptor, hence directly learning from LRF improves the performance of LRF-Net in discriminating correspondences in 3D data.

2.3.3 Generative Learning Models

Generative learning models are approaches that generate output samples by inferring latent representation of input data [33]. As oppose to statistical and discriminative learning models, generative models typically employ discriminative learning as part of its latent representation learning mechanism to learn latent distribution or representation of input data. They are commonly weakly-supervised or unsupervised data-driven approaches and reconstruct close approximation of input samples from learnt latent distribution [74]. Well-established generative learning model on point cloud such as

Auto-Encoder (AE) is an unsupervised architecture comprises an encoder network and a decoder network, where an encoder [73]. An encoder network is optimized to infer the latent representation while the decoder network is optimized to generate output from the latent representation. Early implementation of AE in point cloud representation learning and reconstruction [37] showed remarkable learning generalization and capability in reconstructing output sample that has close approximation to input data. Due to the benefit in generating close approximation of input, generative learning models in point cloud reconstruction are commonly given distorted input such as partial input to generate complete ones. This task is commonly known as point cloud completion.

Due to viewpoint occlusion that is common among depth acquisitions, point cloud completion task have gained momentum in the research field [47]. In a pioneering work in point cloud completion, FoldingNet is proposed by Yang et al.[36] as a grid deformation approach using neural network parametization to mimic the action of folding a 2D grid to match a 3D surface. In FoldingNet, a graph-based encoder is implemented to improve aggregation of local features of input 3D data, while a weight sharing neural network is implemented in decoder network which is more efficient than decoder approach introduced in [37]. However, local details generated from FoldingNet is the drawback of the approach because it "folds" on the shape at global level, hence neglecting local details. To resolve this drawback, AtlasNet [100] is proposed to assemble multiple "folded" part to complete a full shape using efficient shared MLPs. In addition in the computation efficiency advantage, AtlasNet learns at local level, hence it has larger capacity of generalization to novel shape and capability in discriminating shapes.

On the other hand, Point cloud Completion Network (PCN) [34] implemented a graph-based encoder coupled with FoldingNet decoder [36] for point cloud reconstruction. However, the implemented FoldingNet decoder in PCN involves multiple iterative processing that burdens the process to generate output point cloud. On the other hand, Liu et al. [35] proposed a coarse-to-fine grain point cloud sampling by introducing a novel minimum sampling strategy and implemented the patch-based decoder. The method can achieve low reconstruction loss, however the trade-off is realized at the expense of higher network complexity. In a similar fashion, Wang et al. [101] proposed a cascaded refinement network with a coarse-to-fine strategy to synthesize the detailed object shapes. Their framework is designed to keep the object details in the partial inputs and to produce realistic generation of the missing parts using joint reconstruction loss and an end-to-end adversarial loss. While, TopNet [102] proposed a hierarchical tree structure to generate structured point cloud by modeling point cloud topology. This allows better generalization to novel shapes, however the method is intractable to scalability and has limited shapes learning capacity due to the dependency of operating nodes in the tree structure. Moreover, PointGMM [54] is devised to hierarchically learns the topology of 3D shape using neural Gaussian Mixture Model (GMM). The network learns by coarse-to-fine point cloud generation through multiple levels Gaussians.

Besides coarse-to-fine and hierarchical strategies, Wen et al. [103] infer the complete geometries for missing regions of 3D objects using the skip-attention network (SA-Net). They used PointNet++ framework as the backbone of point cloud feature encoder. A skip-attention mechanism is used to effectively exploit the local structure details of incomplete point cloud during the inference of missing parts. A structure-preserving decoder with hierarchical folding block is used to generate complete point cloud. The hierarchical folding preserves the structure of point cloud generated in the upper layer, by progressively detailing the local regions using the skip-attentioned geometric information at the same resolution from the encoder. Furthermore, other researches have implemented novel viewpoint equivariance capsule networks [42, 48, 104, 105] as encoder networks to maximize the discriminative properties of latent features. As a result, the capsule network based generative learning model is more generalized to novel viewpoints and shapes.

2.4 Comparison on 3D Reconstruction Techniques and Discussion

Performance evaluation of the 3D reconstruction techniques are discussed on correspondence accuracy, precision, and recall based on four commonly used dataset, i.e. ModelNet10/40 [28], ICL-NUIM [106] and Semantic3D [107]. The performance metrics associated with non-generative models, i.e. statistical and discriminative learning models are expressed as follows [108]:

(i) Precision is defined as the ratio of correct matches C_{cor} to the correct inlier set C_{in} for classification task. Precision performance implies the capability of a model in recognizing input shape by discriminating the input globally.

$$Precision = \frac{|C_{cor}|}{|C_{in}|} \tag{2.4}$$

(ii) Recall is defined as the ratio of correct or similar shape $_{cor}$ to the ground truth shape C_{GT}. To perform recall efficiently, local 3D feature needs to be concisely encoded to accurately discriminate queried shapes.

$$Recall = \frac{|C_{cor}|}{|C_{GT}|} \tag{2.5}$$

(iii) Correspondence Accuracy (CA) evaluation is defined as the matching of pairwise local correspondences c_1 and c_2 over total matching N.

$$CA = \frac{\sum_N R(c_1, c_2)}{N} \times 100\% \tag{2.6}$$

On the other hand, two widely used dissimilarity metrics for generative models as the reconstruction loss between two sets of point cloud are Chamfer distance (CD) and Earth Mover distance (EMD) [37, 109]. The CD is defined as:

$$d_{CD}(P_1, P_2) = \sum_{x \in P_1} \min_{y \in P_2} ||x - y||_2^2 + \sum_{y \in P_2} \min_{x \in P_1} ||x - y||_2^2, \tag{2.7}$$

where the similarities are simply the distance of two nearest point, thus it is less computational intensive and permutation invariant. However, CD does not consider into surface criterion, such as surface uniformity or surface connectivity [37, 100]. On the other hand, EMD enforces a matching local criterion through a bijection before similarity computation [35]. The EMD $d_{EMD}(\cdot)$ is defined as:

$$d_{EMD}(P_1, P_2) = \min_{\phi: P_1 \to P_2} \sum_{x \in P_1} ||x - \phi(x)||_2, \tag{2.8}$$

where $\phi(\cdot)$ is a bijective function that maps matching P_1 and P_2. Due to EMD's $O(n^2)$ complexity, CD is often chosen for its computation efficiency. Nonetheless, there are several $O(n)$ approximations of EMD such as in [35, 110].

Performance of various 3D representation learning techniques are tabulated in Table 2.1 revealing the precision and recall respectively using benchmark datasets. From the comparisons, generative learning models outperform the discriminative learning models and statistical models by a large margin of precision and recall. The achievement from generative learning models is realized from the advantage of unsupervised learning of the network training. As generative learning models can learn from training data intrinsically, they are more generalized towards novel data. Moreover, significant recall

Table 2.1: Precision and recall performance evaluation using benchmark datasets in local 3D descriptors.

3D Reconstruction Techniques		ModelNet10/40		ICL-NUIM		Semantic3D	
		Precision	Recall	Precision	Recall	Precision	Recall
Statistical models	SDF [82]	28.3/-	25.6/-	-	-	33.2	-
	RoPS [83]	51.6/-	41.3/-	-	-	61.1	-
	SHOT [26]	48.5/-	38.6/-	-	-	57.0	-
	FPFH [25]	40.8/-	34.3/-	52.0	21.7	42.3	-
Discriminative learning models	Srivastava et al. [111]	55.4/-	-	-	-	-	-
	LMVCNN [31]	66.1/-	44.2/-	-	-	66.1	-
	Khoury et al. [94]	65.5/-	48.6/-	-	-	65.6	-
	3DMatch [10]	-	-	65.1	25.2	-	-
	DeepPoint3D [32]	69.5/-	48.8/-	-	-	68.1	-
Generative learning models	Achlioptas et al. [37]	95.4/84.5	-	-	-	-	-
	FoldingNet [36]	94.4/88.4	-	-	-	-	-
	3DPointCapsule [104]	-/89.3	-	-	-	-	-
	Geometric Capsule [42]	96.0/-	94.3/-	-	-	-	-
	Point2Spatial Capsule [105]	95.9/93.7	93.43/89.43	-	-	-	-

Table 2.2: Correspondence accuracy of local 3D descriptors.

3D Reconstruction Techniques		Data Input	Intermediate Data Structure	CA (%)
Statistical models	SDF [82]	Meshes	-	34.8
	RoPS [83]	RGB-D	LRF and RoPS	44.5
	SHOT [26]	RGB-D	Signatures and Histogram	43.1
	FPFH [25]	Point cloud	Point Feature Histogram	36.1
	PIFT [27]	RGB-D	Color coded depth projection	-
Discriminative learning models	Srivastava et al. [111]	Point cloud	Supervoxels	-
	LMVCNN [31]	Point cloud	Projection	63.4
	Khoury et al. [94]	Point cloud	Histogram	60.3
	3DMatch [10]	RGB-D	Voxel Grid	64.7
	DeepPoint3D [32]	Point cloud	-	67.2

performance improvement is observed in capsule based generative learning model. This is because the architecture of capsule network inherits the property of viewpoint equivariance, which has tremendously enhanced the ability of the network to discriminate the shapes with various transformations, such as rotation and translation. Besides, the capsule network architecture also has improved precision over other non-capsule based generative models as the capsule network architecture obtains an advantage in learning part-whole relationship that enable capsule network to deduce whole shape based on observation of parts features.

Table 2.2 shows the summary of data input, data structure and correspondence accuracy between statistical models and discriminative learning models. DeepPoint3D can achieve the highest correspondence accuracy, followed by 3DMatch, LMVCNN, and

CGF. This is due to discriminative learning models which are data-driven and thus are more generalized than statistical model, while also having larger discriminative capability. In a more in-depth experimentation, Srivastava et al. [32] demonstrated a drastic performance degradation in LMVCNN and CGF under low resolution constraint. Such phenomenon could be explained due to the lost information which occurs when 3D data such as RGB-D or 3D point cloud in low resolution is transformed into intermediate representation. However in generative learning models, the networks are capable of directly ingesting raw 3D point cloud, therefore intermediate representation of input data can be avoided in the process. In addition, generative learning models can regenerate its output samples in a higher fidelity, as oppose to statistical and discriminative learning models that are required to search for keypoint correspondences for 3D reconstruction using input data.

Most discriminative learning models prefer to ingest volumetric or voxels grids because it is easier to implement using CNN-based neural networks. However, intermediate input 3D data representation has the issue of rasterization that causes information loss and performance degradation [32, 67]. Therefore, researchers opted to directly process and analyze raw 3D data, i.e. 3D point cloud to further improve the efficacy of computation and minimize information loss. Due to learning capability of DL approaches, discirmative learning models are popularized in learning point cloud to efficiently obtain a rich and condensed representation. However, discriminative learning models can only perform optimally in environment where there is no severe incomplete data, artifacts and viewpoint variances. Thus, generative learning model is devised to extend the learning capability of discriminative learning model to achieve better generalization by learning to generate close approximation output of input data.

2.5 3D Human Modeling and Reconstruction

In conjunction to the advancement in the 3D representation learning and reconstruction techniques, crucial application such as 3D human modeling can be greatly benefited such as gait estimation and tracking for postural assessment [112, 113]. Several existing approaches such as [15, 114–116] have shown success implementation on reconstructing 3D human pose from 2D data. However, they mainly rely on estimating depth to retrieve estimation 3D human pose. One the other hand, several proposed DL networks [34, 35, 102] emerged to reconstruct complete point cloud from partial point cloud that is resulted from single viewpoint acquisition using depth sensor. In conjunction to the advancement in directly reconstruct 3D information from partial data, recent researches [16, 17] are shifting into 3D human pose estimation and human shape using information acquired from depth sensors. In general, 3D human shape reconstruction can be divided into two categories [117], i.e. model-free and model-based 3D reconstruction. The model-free methods do not employ human body models to reconstruct 3D human representation while the model-based methods incorporate parametric body models in the part learning.

2.5.1 Model-free 3D Human Shape Reconstruction

As model-free 3D reconstructions do not take input of human body models with depth information to search for human representation, they predict 3D human pose from 2D images with intermediately estimated 2D pose representation. The common use of deep CNN [15, 114, 115] is deployed for 3D human pose estimation. Li and Chan [114] proposed the use of deep CNN for 3D human pose estimation. Their framework was jointly trained with pose regression and body part detectors to achieve pose projection in 3D space. Tekin et al. [116] combined the traditional CNNs with autoencoder for structured learning to represents the 3D pose from 2D image. A high-dimensional latent pose representation learned by autoencoder was reprojected to original pose space with a decoding layer to account for joint dependencies.

Besides that, implicitly learning of the pose structure from 2D data has recently drawn into attention to infer 3D human pose with two separate sequential training steps. They first perform 2D joint prediction and then reconstruct the 3D pose via optimization or search. Pavlakos et al. [15] applied ConvNet for 2D joint location and subsequently perform optimization step to recover 3D pose. They used volumetric representation for 3D human pose and employed a coarse-to-fine prediction scheme to do refinement in 3D pose estimation. To improve the 3D ground truth accuracy, they used the ordinal depths of human joints as the supporting signal to perform weakly supervision in the 3D human pose learning [118]. Sun et al. [119] proposed the use of bones instead of joints as pose representation. Subsequently, it exploited the joint connection structure to define a compositional loss function that encodes long range interactions between the bones.

Zhao et al. [120] pre-trained a 2D pose estimation network to predict 2D joint locations. Subsequently, a semantic graph convolutional network is trained to predict

3D pose from 2D joints features. Cheng et al. [14] proposed an occlusion-aware DL framework to estimate a 2D confidence heatmaps of keypoints. With the optical flow consistency constraint, unreliable estimations of occluded keypoints were filtered and subsequently fed into 3D temporal convolutional networks to produce a complete 3D pose. Xu et al. [121] performed deep kinematics analysis using 2D noisy pose inputs to obtain 3D pose estimation concurrently by considering the static and dynamic body structures. The limitation of model-free 3D reconstruction is restricted by its accuracy of depth estimation from sensors because the captured data contains numerous artifacts such as occlusion, outliers and non-uniform surface.

2.5.2 Model-based 3D Human Shape Reconstruction

While Model-free approaches are relying on estimation of depth from 2D images, hence the predicted 3D pose can be unreliable due to estimation process. Moreover, Model-free approaches do not provide expressive visual output of 3D human such as parametric body shapes. The Model-based 3D reconstruction [122–126] on the other hand incorporates parametric body shapes such as body pose and body volume to perform 3D human reconstruction. Widely used volumetric models for synthetic human body construction are Fine Alignment Using Scan Texture (FAUST) model [122], Skinned Multi-Person Linear (SMPL) model [123]. Litany et al. [127] proposed the use of variational autoencoder incorporated with graph convolutional operations for the completion of human shape reconstruction. It learns a latent space for complete realistic shape with vertex-wise correspondence using FAUST synthetic dataset. In recent works, SMPL model is commonly used to perform 3D parameter estimation to construct a full human body. Several extended SMPL-based models such as SMPLify [124] and Vposer [125] are devised to reconstruct 3D human model through skeletal joints regression. The latter, Sparse Trained Articulated Human Body Regressor (STAR) [126] is introduced with improvement over SMPL by training with additional 14,000 human subjects and a learning set of sparse local pose corrective blend shapes. In addition, the number of parameters in STAR is reduced to 20% of that in SMPL model.

Kinematic model in 3D space has gained increasing attention in 3D human pose estimation recently because it is a realistic and accurate articulated body representation. By using single depth sensor, Zhou et al. [17] infers 3D joint positions from partial point cloud with a 3D pose regression network without 3D human reconstruction. In recent trend, kinematic and synthetic 3D human models are used to enhance 3D human reconstruction. For example, Jiang et al. [16] proposed to incorporate skeleton joints into a DL network for 3D human shape reconstruction. The basic structure of this model uses PointNet++ to extract point features and then map point features to skeleton joint features and finally SMPL parameters for the 3D human reconstruction. In general, SMPL offers a simple integration and compact representation for 3D human reconstruction. Overall, Model-based can reconstruct an expressive output of 3D human, however most existing methods that reconstruct a 3D human are inferring on complete point cloud rather than partial point cloud. Hence, several aspects in 3D human shape reconstruction are covered using recent research trend in generative learning models such as skeletal joints inference and shape reconstruction from input partial point cloud.

2.6 Chapter Summary

The process of 3D modeling is heavily dependent on the 3D reconstruction techniques, which can be widely divided into three categories, i.e. statistical models, discriminative learning models and generative learning models. Statistical models analyse the input data and extract feature descriptor for maximum likelihood matching. These models are computational efficient, however perform poorly when dealing with novel data and generalization. On the other hand, discriminative learning models use supervised learning approach to establish local descriptor that is driven by labeled data training. These models allow higher learning capability in establishing descriptors compared to statistical learning models and can perform better in task-oriented objective and novel data generalization. In the advent of generative models, raw 3D point cloud are ingested directly into the network which can improve the efficacy of point cloud reconstruction process. Without the data pre-processing and post-processing, generative models could generalize the data learning by learning the latent representation of input data and subsequently generate close approximation output of input data. As generative models generate output samples from learnt latent representation, they are robust to noises such as outliers. In performance evaluation of generative models, they outperformed other models in both precision and recall with significant margin. They are also able to learn exceptionally well in differentiating novel shapes with arbitrary viewpoint transformations using part-whole relationship via unsupervised learning. As of current research trend, generative learning models would be the main architecture in point cloud reconstruction techniques due to its advantages. Research effort on view-point invariant reasoning from local geometrical features could be emphasized to fully exploit the sparseness and unstructured nature of 3D data. In addition, learning from local geometrical feature could also introduce a new edge of fidelity of output samples. Led by the advancement in generative learning models, applications such as point cloud classification, reconstruction and 3D human shape reconstruction are able to achieve new milestone in their performances. For instance, skeletal joints inference can be closely related to point cloud representations learning, while 3D human shape reconstruction can be closely related to point cloud reconstruction, and they can greatly benefits from the advantages from generative learning models. In the following chapters, novel 3D point cloud representation learning and reconstruction models are proposed and application on 3D human modeling is showcased using the proposed model.

Chapter 3

Point Cloud Neural Density Estimation

3.1 Introduction

Point cloud is a collection of data points in sparse and grid-less space, typically (3D) coordinate space to represent an object. It is more expressive and contains higher geometric detail than voxel grids [10, 29, 30]. Also, point cloud is a stepping stone in achieving more precise representation of an object model such as meshes [128, 129]. Using a generative model to learn point cloud representation is beneficial to a wide range of 3D tasks such as reconstruction [34, 100, 104] and classification [4, 20, 52]. Besides that, statistical models [130, 131] can be modelled after the point cloud distribution in obtaining a condensed representation of a generic 3D shape. However, processing point clouds possess challenging issues due to its properties such as the sparsity and irregularity of data modality in 3D space. The irregular sampling of point clouds requires a better centered object representation that can provide surfaces with arbitrary complexity and topology [132].

The capability to process irregular and un-ordered point clouds using neural networks is first introduced in PointNet [4] due to its point-wise convolutions and the permutation invariant symmetric functions. However, point-wise operations do not correlate with neighbourhood points. Subsequently, PointNet++ [7] is proposed as a hierarchical neural network to combine sampling layer, group layer and PointNet layer to capture multi-scale patterns. Many research works [52, 133, 134] have extended Point-

Net primarily to increase the size of local receptive field to explore the relationship of neighbourhood points. Unfortunately, the convention convolution operators are solely invariant to permutations and translations. The introduction of spherical kernel CNN and capsule networks [48, 135, 136] learn a rotation equivariant representation of a 3D shape with complex learning mechanisms. Recent efforts [56, 57] have moved towards unsupervised learning in the point clouds processing models with latent variables corresponding to the variation in the data. The unsupervised models such as Autoencoder [34, 37, 100] and Generative Adversarial Network (GAN) [137, 138] are used to generate the data distribution implicitly to represent an object in terms of sampling pattern and pose invariance.

Inspired from the mechanism of generative models, a point cloud density estimator is proposed in this chapter to generate a set of multivariate Gaussian mixture components to represent density point cloud. Conventionally, iterative estimation of mixture coefficient is required to model a mixture model. In this work, latent inference using neural network is adopted to resolve the iterative term. By obtaining the density parameters of a point cloud, local patch of an object can be semantically segmented, i.e. clustering, to represent a local part of the object. The proposed network has two stages of network modules: (i) Global Density Encoder and (ii) Local Density Autoencoder. Essentially, the formulation of the proposed network is to achieve latent inference by first, generating a latent global density, and subsequently inferring and projecting the latent local density from the latent global density into ambient density parameters. The latent inference is devised such a way the latent variable is optimized to satisfy the maximization of the objective function. Through latent inference, a mixture model can be estimated directly by maximizing the log-likelihood. As the density parameters estimation is a condensed representation of data structure of a point cloud, it can be applied for tasks, such as point cloud classification and reconstruction.

3.2 Theory of Density Estimation

Density estimation [130, 131] is used to address one of the most fundamental problems occurred in machine learning, which is the problem of self-discovering structure from data in an unsupervised manner. In definition, a density function $q(x)$ is defined as an estimate description of the joint statistical distribution of data samples x. Therefore, it is commonly associated in models of data sampling [55, 138], where tasks such as novel data generation and data prediction can be achieved by drawing samples from the density function. In contrast, density estimation is the reverse of sampling process, where the density function is retrieved by observing data samples from which the samples are assumed to be generated from a true density function $p(x)$. In this work, the concept of Bayesian inference and density estimation are applied to learn the density parameters that represents the distribution structure of a point cloud using a parametric mixture model.

3.2.1 Bayesian Inference

In Bayesian framework, the Bayes' rule encodes the belief in a parameter ϕ that express a data structure and changes the degree of belief in ϕ based on newly observed data. Typically, ϕ is set as a unknown random variable with a density function $p(\phi)$, which is the prior belief about an expression of a set of unobserved data x. From Bayes' rule, the degree of belief in ϕ changes when x is observed as new evidences and the changes are commonly made based on a statistical model $p(x|\phi)$ that reflects the belief about x given ϕ. Therefore, the statistical relationship between x and ϕ is established as follows,

$$p(\phi|x) \;\propto\; p(x|\phi)\; p(\phi), \tag{3.1}$$

where $p(\phi|x)$ is the posterior of updated belief in ϕ when given x and $p(x|\phi)$ is the conditional likelihood of x given ϕ. In the context of this chapter, the Bayesian inference is related to the training of point clouds density estimation using a neural parametric mixture model. The goal is to use a neural network to estimate the best setting of ϕ given x, which can be viewed as a Maximum A Posteriori (MAP) estimation:

$$\hat{\phi}_{MAP} = \underset{\phi}{\operatorname{argmax}}\, p(x|\phi)\; p(\phi), \tag{3.2}$$

where $\hat{\phi}$ gives the maximum likelihood of observed data. Thus, by obtaining $\hat{\phi}$ an estimated density function can be estimated as a close approximation of the true density $q(x|\hat{\phi}) \simeq p(x)$. In application of density estimation, uniform prior can be typically chosen for a model's setting to achieve estimated posterior. Hence, the MAP of density parameters can be simplified as a Maximum Likelihood Estimation (MLE) as follows,

$$\begin{aligned}\hat{\phi}_{MAP} &= \underset{\phi}{\operatorname{argmax}}\, p(x|\phi)\; p(\phi) \\ &= \underset{\phi}{\operatorname{argmax}}\, \ln(p(x|\phi)) + \ln(p(\phi)) \\ &= \underset{\phi}{\operatorname{argmax}}\, \ln(p(x|\phi)) + \text{const} \\ &= \hat{\phi}_{MLE}.\end{aligned} \tag{3.3}$$

3.2.2 Parametric Mixture Model

The process of a simple parametric density estimation typically focuses in finding the best setting of a density parameter to obtain maximum likelihood from the observed data. In this implementation, multivariate Gaussian distribution is used for parametization of the estimated density function $q(x|\phi)$ and the marginal likelihood is defined as follows,

$$q(x|\phi) = \prod_i \mathcal{N}(x_i|\phi), \tag{3.4}$$

where

$$\mathcal{N}(x_i|\phi) = \frac{1}{|\det(2\pi\Sigma)|^{1/2}} \exp\left\{-\tfrac{1}{2}(x_i - \mu)^T \Sigma^{-1} (x_i - \mu)\right\}, \tag{3.5}$$

$\phi = \{\mu, \Sigma\}$ denotes a Gaussian component. $\mu \in \mathbb{R}^D$ denotes the mean and $\Sigma \in \mathbb{R}^{D\times D}$ denotes the covariance matrix with dimension of D. In order to expand density representation capability from a simple parametric model, a Gaussian Mixture Model (GMM) comprises M Gaussians is devised to capture more localized details, i.e. clustering, on an observed data and is defined as follows,

$$q(x|\pi, \phi) = \prod_i \sum_{m=1}^{M} \pi_m \mathcal{N}(x_i|\phi_m), \tag{3.6}$$

where $\phi = \{\phi_1, \ldots, \phi_M\}$ are the GM components, $\sum_{m=1}^{M} \pi_m = 1$ denotes the mixing coefficient and $\pi_m \geq 0$ for all m. To find the best estimates of GM components, the optimization can be achieved by directly maximizing the average log-likelihood. However, solving the maximum likelihood of a GMM is non-trivial due to the summation in log defined as follows,

$$\ln(q(x|\phi)) = \sum_i \ln \sum_{m=1}^{M} \pi_m \mathcal{N}(x|\phi_m), \tag{3.7}$$

Commonly, an iterative procedure such as Expectation-Maximization (EM) algorithm is employed to fit the GMM by re-estimating the posterior probability of cluster assignments. Alternatively, a latent variable α can be introduced in a mixture model to simplify Eq. (3.7) as follows,

$$q(x, \alpha|\pi, \phi) = \prod_{i} \prod_{m=1}^{M} \pi_m^{\alpha_{i,m}} \mathcal{N}(x_i|\phi_m)^{\alpha_{i,m}}, \tag{3.8}$$

where $\alpha_{i,m}$ is a one-hot encoding of M density assignment. The maximum log-likelihood is then derived as follows,

$$\ln(q(x, \alpha|\pi, \phi)) = \sum_{i} \sum_{m=1}^{M} \alpha_{i,m}\{\ln(\pi_m) + \ln(\mathcal{N}(x_i|\phi_m))\}. \tag{3.9}$$

3.3 Point Cloud Neural Density Estimator

Assuming that a generative sampling process produces a 3D point clouds dataset $x = \{x_i \in \mathbb{R}^{3\times N}, i = 1, 2, ..., S\}$, where S is the total number of point cloud samples and N is the number of points in 3D space with X-Y-Z coordinate. Assuming a joint density exists in a point cloud containing a set of M density parameters, such that $x_i \sim \mathcal{N}(\phi)$, where $\phi = \{\phi_1, \ldots, \phi_M\}$, and each Gaussian component is denoted as $\phi_m = \{\mu_m, \Sigma_m\}_{m=1}^{M}$. Hence, the dataset of 3D point clouds can be statistically represented with a mixture model. Suppose a point cloud can be deconstructed into M local parts $\hat{x}_{i,m} \in \mathbb{R}^{3\times \frac{N}{M}}$, i.e. local patch clustering, where a point cloud $x_i = \{\hat{x}_{i,1}, \ldots, \hat{x}_{i,M}\}$ is a collection of the M local parts. Therefore, a local patch is defined to be independently

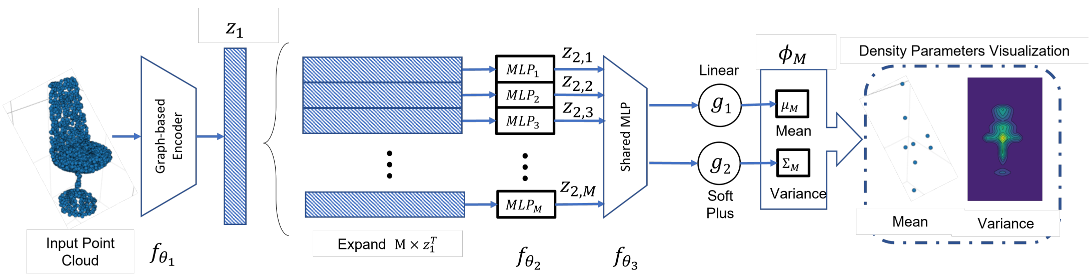


Figure 3.1: The proposed Point cloud Neural Density Estimator composes of Global Density Encoder $f(\mathcal{G}, \theta_1)$ and Local Density Autoencoder $f(\theta_2, \theta_3)$ to generate density parameters of point clouds mixture model. Linear and Softplus activation functions g_1 and g_2 is applied to the last layer of the model. The density parameters are visualized by plotting the mean and average variance of a point cloud of chair.

sampled from an independent Gaussian component or local density distribution $\hat{x}_{i,m} \sim \mathcal{N}(\phi_m)$.

Neural density estimation is a parametric method to estimate a density function using a deep neural network. A neural network $f(\theta)$ as a universal function approximator contains a set of trainable parameters θ that can be tuned accordingly to an objective function, i.e. maximum log-likelihood. Using intuition from the parametric mixture model in Eq. (3.4), a neural network can be flexibly devised to estimate best setting of $\hat{\phi}_{MLE} = f(\theta)$. In an existing work, PointGMM [54] learns to generate multiple levels of Gaussians, where top-level Gaussian covers larger region of shape and low-level Gaussian covers local surface details. Intuitively, the proposed network in this chapter aims to estimates fully Gaussians by single pass. Inspired from the setup of neural density estimation architecture in [139, 140], an encoding function is used to estimate the cumulative distribution function (CDF) to obtain a global distribution. Subsequently, the probability distribution function (PDF) is retrieved by a reverse sampling process from the CDF. Therefore in this work, Point cloud Neural Density Estimator (PNDE) is proposed as shown Figure 3.1 to address the iterative parameters, i.e. the mixture coefficient π and the cluster assignment α by latent inference to allow training of a PNDE by maximizing the log-likelihood. The PNDE comprises two network modules: (a) Global Density Encoder, and (b) Local Density Autoencoder.

3.3.1 Global Density Encoder

Similar to AE [37] and VAE [141], latent features are commonly first encoded which can provide several advantages such as reduced dimension during inference and the unrolling and disentanglement of latent variations in the input data [130]. Another important criteria in point cloud latent feature encoding is the property of permutation invariant to adapt the nature of un-ordered points in point cloud. Generally, a point cloud latent feature encoder can be categorised into two types, i.e. point-wise [4] and graph-based [52]. Due to the higher capability in graph-based encoder to encode point cloud latent features by incorporating the notion of neighbourhood points, it is adopted as the Global Density Encoder (GDE) $f(\mathcal{G}, \theta_1) : \mathbb{R}^{3\times N} \mapsto \mathbb{R}^d$, where d is the dimension size of the latent global density to encode the latent global density $z_1 \in \mathbb{R}^d$ of a point cloud x_i defined as follows,

$$z_1 = f(\mathcal{G}, \theta_1; x_i). \tag{3.10}$$

The GDE comprises a graph feature extraction module $\mathcal{G} : \mathbb{R}^{3\times N} \mapsto \mathbb{R}^{6\times N\times k}$ and an embedding network $f_{\theta_1} : \mathbb{R}^{6\times N\times k} \mapsto \mathbb{R}^{d'}$. The feature extraction strategy is inspired from [52] by constructing directed k-nearest neighbour (k-NN) graphs $\mathcal{G} = (V, E)$. The vertices $V = \{v_0, v_1, ..., v_N\}$ of an input point cloud are connected by edges $E = \{e_{ij_1}, e_{ij_2}, ..., e_{ij_k}\}$, where the edges are computed based on k-smallest oriented pair-wise distance, such that:

$$e_{ij_k} = ||v_{j_k} - v_i||^2, \tag{3.11}$$

or equivalent to vectorized pair-wise squared Euclidean distance from Eq. (3.11), that is optimized for parallelism in GPU computation [142]:

$$E = \{e_{ij} \in \big(||V||^2 - 2V^T V + ||V^T||^2\big)\}_k. \tag{3.12}$$

3.3.2 Local Density Autoencoder

Subsequent from obtaining the latent global density, Local Density Autoencoder (LDA) is devised to output local patch density with an encoder module $f(\theta_2)$ and a decoder module $f(\theta_3)$. The overall LDA network mapping is then defined as $f(\theta_2, \theta_3) : \mathbb{R}^d \mapsto \mathbb{R}^{6\times M}$ and it infer latent density to obtain the local patch cluster. The implementation of latent inference in LDA is to address the issue of iterative cluster assignment and unknown mixture coefficient. Intuitively, the cluster assignment parameter α is defined as a one-hot encoding that defines the assignment of likelihood with respect to the most probable density. To achieve this, a fixed number of independent density sampling function as an encoder module can be introduced to explicitly define the assignment of a cluster [140]. Thus, by taking advantage of independent density estimation of each density parameter, the cluster assignment procedure can be systematically assigned to designated encoder module.

Therefore, the encoder $f(\theta_2) : \mathbb{R}^d \mapsto \mathbb{R}^{d'\times M}$ comprises a set of independent weights $\theta_2 = \{\theta_{2,1}, \dots, \theta_{2,M}\}$ is implemented and it is equivalent to an independent density sampling function. Secondly, the mixing coefficient is viewed as a prior knowledge in soft-clustering a set of data points into respective cluster. With latent inference, the mixing coefficient is implicitly encoded in the latent local density, such that $z_{2,m} \sim \mathcal{N}(z_1|\theta_{2,m}^{\pi_m,\alpha_{i,m}})$. Hence, the latent inference can be viewed as drawing the encoded local densities $z_2 = \{z_{2,1}, \dots, z_{2,M}\}, z_2 \in \mathbb{R}^{d'\times M}$ from the latent global density defined as

Table 3.1: Comparison of parameters and operations of independent MLP and superimposed MLP.

MLP	Parameters configuration	Operation
Independent	$(d \times d' \times M)$	$z_2 = \{\theta_{2,m} \odot z_1\}_{m=1}^{M}$
Superposed	$(d \times d') + (d \times M) + (d' \times M)$	$z_2 = ((z_1 \odot \theta_{2,s})\theta_{2,w}) \odot \theta_{2,r}$

follows,

$$z_{2,m} = f(\theta_{2,m}^{\pi_m,\alpha_{i,m}}; z_1), \tag{3.13}$$

where the network parameter $\theta_{2,m}^{\pi_m,\alpha_{i,m}}$ is explicitly assigned to m-th cluster and implicitly containing the mixture coefficient. Due to the implementation of independent weights in the encoder module, the number of network parameter scales exponentially when M is set to a large number. Moreover, an iterative function is typically required to iterate over the list of parameters. To resolve these concern, a parameters compression strategy [143] is adopted to reduce the number of parameter. In addition, matrix vectorization and decomposition is incorporated along the compression strategy to allow decomposition of $\theta_2 \in \mathbb{R}^{d \times d' \times M}$ into $\{\theta_{2,w} \in \mathbb{R}^{d \times d'}, \theta_{2,r} \in \mathbb{R}^{d' \times M}, \theta_{2,s} \in \mathbb{R}^{d' \times M}\}$ as shown in Table 3.1. Essentially, network configuration without matrix vectorization and weight superposition can cost tremendous forward propagation time due the iterative inference from independent MLPs. Whereas, with implementation of matrix vectorization and weight superposition, the inference can be computed in several matrix computation. Finally, a shared weight network as a decoder network $f(\theta_3) : \mathbb{R}^{d'} :\mapsto \mathbb{R}^{6 \times M}$ is devised to project the latent local densities into ambient density parameters ϕ as follows,

$$\phi = \{f(\theta_3; z_{2,1}), f(\theta_3; z_{2,2}), ...f(\theta_3; z_{2,M})\} \tag{3.14}$$

3.3.3 Objective Function

By aggregating GDE and LDA, a PNDE with the mapping $f(\mathcal{G}, \theta_1, \theta_2, \theta_3) : \mathbb{R}^{3 \times N} \mapsto \mathbb{R}^{6 \times M}$ is formulated to fit a mixture model in Eq. (3.15) to estimate the density parameters of a point cloud.

$$\begin{aligned}\ln(q(x, \alpha|\pi, \phi)) &= \ln(q(x|f(\theta))) \\ &= \sum_{i}^{S} \sum_{m=1}^{M} \ln(\mathcal{N}(x_i|f(\theta)),\end{aligned} \tag{3.15}$$

To allow the network model undergoing the training in a self-supervised manner, a network's objective function is required to guide the parameter learning process. The objective function of PNDE is defined as,

$$\begin{aligned}\mathcal{L}_{PNDE}(x, \phi) = \frac{1}{S}\frac{1}{N}\frac{1}{M}\sum_{i=1}^{S}\sum_{j=1}^{N}\sum_{m=1}^{M} \\ \left(-\frac{3}{2}\ln(2\pi) - \frac{1}{2}\ln(|\Sigma_m|) - \frac{1}{2}(x_{i,j} - \mu_m)^T\Sigma^{-1}(x_{i,j} - \mu_m)\right).\end{aligned} \tag{3.16}$$

where S is the number of sample size in a dataset, N is the number of points in a point cloud, and M is the number of density parameter. As the objective function of PNDE is boiled down to maximization of the average log-likelihood, it is equivalent in minimizing the KL-Divergence [131] between $p(x)$ and $q(x, \alpha|\pi, \phi)$):

$$D_{KL}(p(x)||q(x, \alpha|\pi, \phi)) = -\mathbb{E}_{p(x)}(\ln q(x|f(\theta)))) + \text{const} \tag{3.17}$$

Using the strong law of large numbers of samples as $S \mapsto \infty$, $\mathcal{L}(\theta)$ can converge almost to $\mathbb{E}_{p(x)}(\ln q(x|f(\theta))))$.

3.4 Experiment Setup

Performance of the proposed model is evaluated using three benchmark 3D datasets, i.e. ModelNet10/40 [144] and ShapeNet [145] dataset. The ModelNet40 contains 40 categories of labelled CAD models respectively from common objects with 9843 training and 2468 test data. The ModelNet10 is a subset of ModelNet40 which contains 10 categories with 3991 training and 908 testing data. Each object is normalized into unit sphere $[-1, 1]^3$. On the other hand, ShapeNetCore13 is customized as a subset of 13 categories out of 55 categories from ShapeNetCore with 31,772 training and 7,956 testing data. Data augmentation during training includes only points permutation in the input point clouds.

Table 3.2: Configuration of GDE, LDA, and classification network.

Layers	Configuration	Output Dim.
Input	-	$N \times 3$
GDE	EdgeConv ($k = 10$) $[3, 64, 128, 128]$	$N \times 128$
	Conv1D $[128, 256]$ +BN+ReLU	$N \times 256$
	Conv1D $[256, d]$ +BN+ReLU	$N \times d$
Pooling	Maxpool1D (d)	$1 \times d$
LDA	Superposed MLP $(d \times d') + (d \times M) + (d' \times M)$ +BN+ReLU	$M \times d'$
	Conv1D $[d', d'']$ +BN+ReLU	$M \times d''$
	Conv1D $[d'', 6]$ +BN	$M \times 6$
GM Components	ReLU ($M \times 3$)	$M \times 6$
	Softplus ($M \times 3$)	
-	Flatten	$1 \times 6M$
Classification network	FC layer $6M \times 512$ +BN+ReLU	1×512
	Dropout (0.3)	-
	FC layer 512×256 +BN+ReLU	1×256
	Dropout (0.3)	-
	FC layer $256 \times C$ +Softmax	$1 \times C$

Three experiments are set up to evaluate the proposed PNDE. Experiment #1 evaluates qualitative results of the density parameters estimated from the PNDE, effects of hyperparameters d, d', M and overall classification accuracy on ModelNet40. Experiment #2 evaluates classification accuracy of the PNDE on ModelNet10 and ShapeNet-Core13 using classification network in [4]. The architectural details of network setup is shown in Table 3.2. In each experiment, PNDE is pre-trained using the preset hyperparameters $N = 2048$, $d = 3, d' = 1024, d'' = d'/2 = 512$, and number of Gaussian components $M = \{8, 16, 32, 64\}$. The number of Gaussian components are chosen as multiple of 8 to fully utilize the GPU computation. Training of PNDE involves 100 epoch with batch size 64 and learning rate of 10^{-3} with scheduling decay rate of 0.5 per 20 epoch. ADAM optimizer is used as optimization method to maximize the log-likelihood. In experiments #1 and #2, the classification network is trained for 100 epoch with scheduling decay rate of 0.5 per 20 epoch using number categories as the batch size.

3.5 Experiment #1: Point Clouds Neural Density Estimator

In this experiment, a classification task is setup to evaluate the performance on ModelNet40. In Table 3.3, the overall accuracy for classification using $M = \{8, 16, 32, 64\}$

achieved $88.32\%, 87.56\%, 88.57\%, 88.74\%$ and 89.26% respectively on ModelNet40. With representation using 64 GM components, the proposed PNDE achieved the best classification rate (88.74%) and is comparable to other methods with parameter count of 0.90M + 0.34M (0.90M is the total number of learning parameters in the proposed PNDE, and 0.34M parameters are referred to classification network).

In Table 3.4, the experiments are conducted using hyperparameters $N = 2048$, $M = 64$ and with varied hyperparameters $d = \{128, 256, 512, 1024\}$ to test the robustness of PNDE. The results show the overall classification accuracy with respect to model size of the proposed network by varying GDE bottleneck size d. Overall classification accuracy of 87.44% - 88.74% is achieved with respect to $d = \{128, 256, 512, 1024\}$, and 87.44% with the minimum network parameter of 0.06M + 0.34M.

Fig. 3.2 shows the qualitative results of the estimated density parameters of various objects using $M = 32$. In Fig. 3.3, an averaged variance is projected as the contour on XY, YZ, and XZ plane for variance visualization. The visualization of variances indicates the learning of point clouds distribution in each density. Furthermore, the visualization also shows the multivariate variance effectively distributed the projected

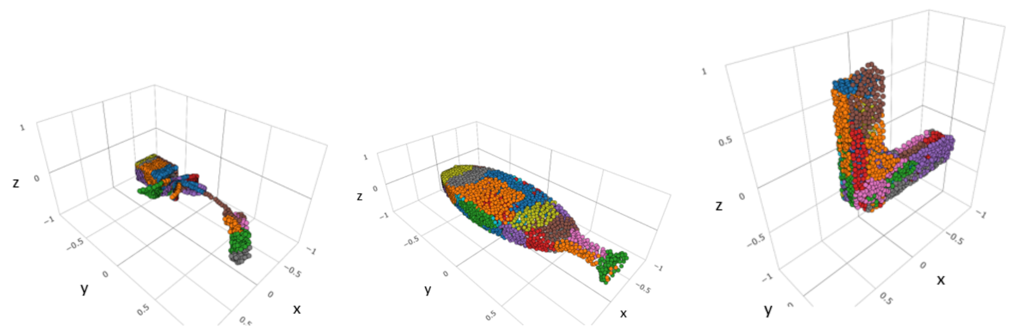
Table 3.3: Overall accuracy (OA) of PNDE on classification task. #param is the network parameter excluding the parameter in MLP classifier with $d = 1024$.

M	Bottle-neck	#param	OA (%)
8	48	0.82M	87.32
16	96	0.83M	87.56
32	192	0.85M	88.57
64	384	0.90M	88.74

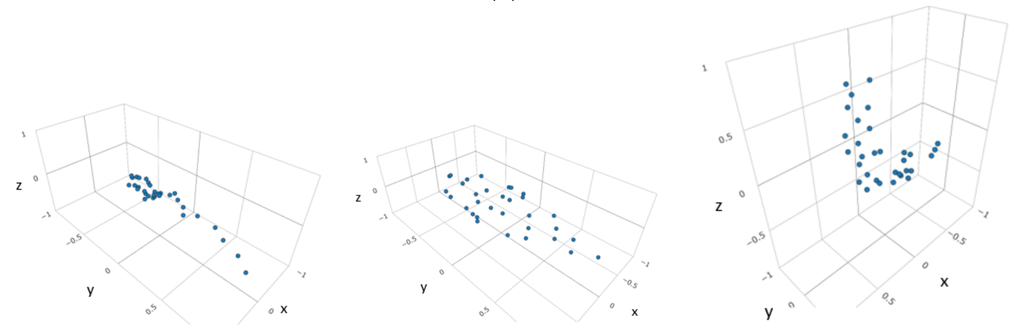
Table 3.4: Comparison of overall accuracy (OA) of point clouds classification method with $M = 64$ and varied GDE bottleneck d.

Configuration	*#param	OA (%)
$d = 128$	0.06M	87.44
$d = 256$	0.12M	87.88
$d = 512$	0.30M	88.49
$d = 1024$	0.90M	88.74

* Additional 0.34M parameter is added for MLP classifier

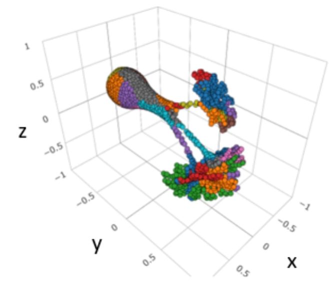


(a)

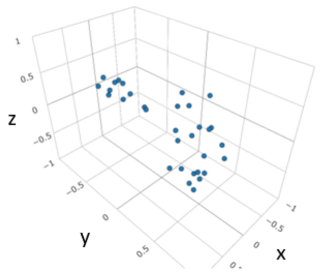


(b)

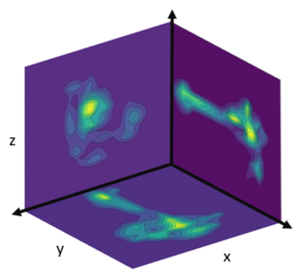
Figure 3.2: Qualitative results of PNDE with $M = 32$ density parameters: (a) M clusters of color-coded local patch point cloud and, (b) Mean of each local patch point cloud.



(a)



(b)



(c)

Figure 3.3: Qualitative results of point cloud distribution using contour map visualization: (a) Color-coded clustering in ground truth point cloud, (b) Estimated mean from ground truth point clouds, and (c) Average variance $(\Sigma^2_{xy}, \Sigma^2_{yz}, \Sigma^2_{xz})$ projected on XY, YZ and XZ plane of $M = 32$.

shape of input point clouds with respect to the mean. To show point clouds representation different number of density parameters, Fig. 3.4 visualizes mean representation of point clouds using with $M = \{8, 16, 32, 64\}$.

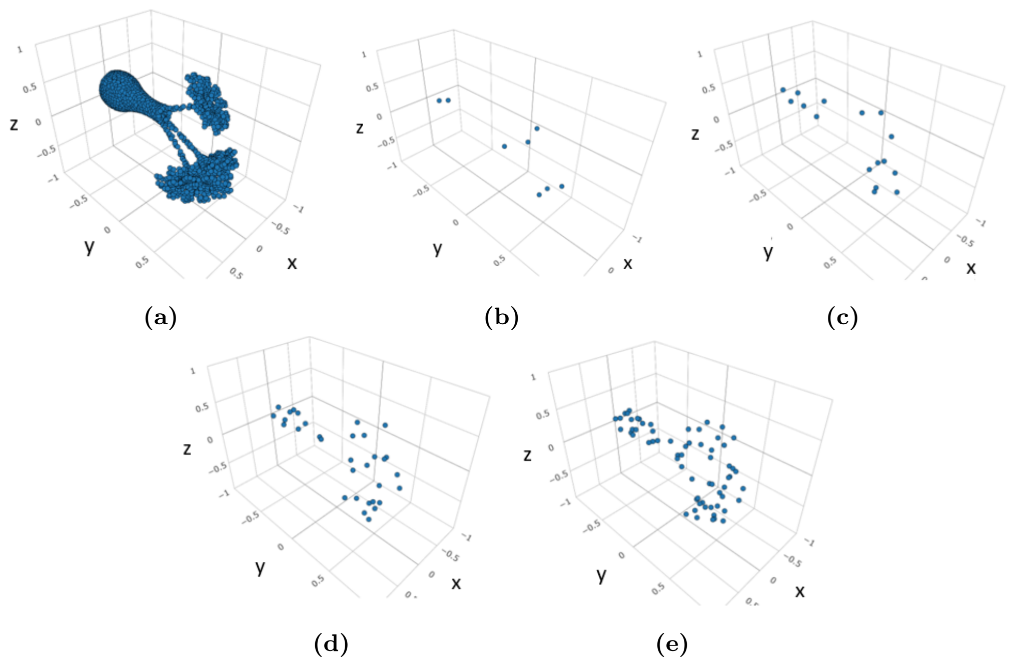


Figure 3.4: Qualitative results of PNDE with M density parameters: (a) Ground truth point clouds, (b) $M = 8$ (c) $M = 16$, (d) $M = 32$, (e) $M = 64$.

3.6 Experiment #2: Point Cloud Classification

To further evaluate the proposed network, the classification task is extended to other dataset i.e. ModelNet10 and ShapenetCore13. Table 3.5 shows comparisons between some of the popular existing works as compared to the proposed PNDE. The baseline in the classification task is PointNet [4] and DGCNN [52] due to their pioneering state-of-the-art performance in the field. The evaluation metrics of PointNet and DGCNN are reproduced using the available source code with a configuration stated in their implementation.

PointNet is a point-wise learning network that focuses on global feature, and PointNet++ is the extension of PointNet that focuses on local feature. Both ECC and DGCNN share similarity in using edges of neighbourhood vertices as edge features, which is also the feature learning strategy implemented in the proposed network. Meanwhile, spherical CNN used a specialized local feature kernel (spherical kernel) and graph convolution. As spherical CNN and DGCNN have a higher network complexity compared to the proposed model, they naturally contain more capacity in discriminating local features, thus perform better in classification.

Overall, the classification rate implies that the estimated density parameters are

Table 3.5: Comparisons of overall accuracy (OA) of point clouds classification on Model10/40 and ShapenetCore13 dataset.

Method	#point	Bottleneck	#param	OA (%)		
				ModelNet10	ModelNet40	ShapeNetCore13
PointNet [4]	2048	1024	3.5M	92.75	89.4	94.7
ECC [133]	1000	64	0.2M	90.0	83.2	-
DGCNN ($k = 10$) [52]	2048	1024	1.8M	94.5	91.5	95.1
Spherical CNN ($k = 64$) [136]	2048	832	0.7M	-	91.4	-
Ours ($k = 10, M = 64$)	2048	384	1.24M	93.67	88.74	94.60

effective in representing input point clouds using significantly fewer number of features. Is is also a flexible network where M can be tuned as hyperparameter to produce more density parameters. On the other hand, Spatial Transformer Networks (STNs) are implemented in baseline methods to encounter input data augmentation (e.g. rotation) by approximating an orthonormal transformation matrix to compensate input point clouds into canonical representation. However, STN can act as an external transformation that disrupt rotational equivariance in a network.

3.7 Experiment #3: Rotation Equivariance Evaluation

Definition 1 (Rotational Equivariant Network) For a network to be rotational equivariant, all function map in a network must be an equivariant map and the resultant rotation in the input should cause equivalently transformation in the output [146, 147], such that:

$$\Lambda(g_1 \circ \mathcal{X}) = g_2 \circ \Lambda(\mathcal{X}), \tag{3.18}$$

where $\{g_1, g_2\} \in \mathcal{G}$ and $\mathcal{G}$ is affine transformation group. Λ is an equivariant transformation function map and $\mathcal{X}$ is data group.

Definition 2 (Rotational Equivariant Local Maximizer) A density estimator is equivariant under linear affine transformation, e.g. rotation. Given a random variable $\mathcal{X}$ is sampled from μ and Σ:

$$\mathcal{X} \sim \mathcal{N}(\mu, \Sigma). \tag{3.19}$$

Let $\hat{\mathcal{X}} = A\mathcal{X} + b$ be an affine transformation acted on $\mathcal{X}$, then $\hat{\mathcal{X}}$ can be sampled from density parameters of $\hat{\mu}$ and $\hat{\Sigma}$:

$$\hat{\mathcal{X}} \sim \mathcal{N}(\hat{\mu}, \hat{\Sigma}). \tag{3.20}$$

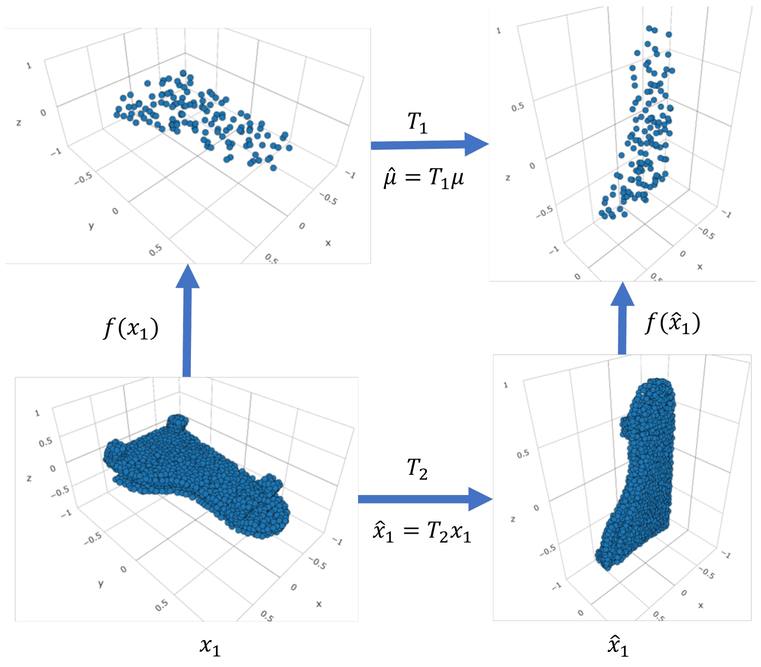


Figure 3.5: Qualitative results of the PNDE demonstrating rotational equivariance by inferring the mean of density parameters, μ and $\hat{\mu}$ of an input point cloud x_1 and a rotated point cloud $\hat{x}_1$ under transformation $T_1 \simeq T_2$.

where $\hat{\mu} = A\mu + b$ and $\hat{\Sigma} = A\Sigma A^T$. Let $\mathcal{N}(\mu, \Sigma|\mathcal{X})$ be the local maximizer for $\mathcal{X}$, then $\mathcal{N}(\hat{\mu}, \hat{\Sigma}|\hat{\mathcal{X}})$ is the local maximizer for $\hat{\mathcal{X}}$ [148].

Transformation equivariant is essential in the analysis of point cloud, because it preserves the original state and orientation from the input. This is crucial as the point cloud needs to retain its identity regardless of an affine transformation. Therefore, by using **Definition 1** and **Definition 2**, let $\{T_1, T_2\} \in \mathcal{T}$ act on x_i, the PNDE $f(\theta)$: $\mathcal{T} \times x_i \mapsto [\hat{\mu}, \hat{\Sigma}]$ is rotational equivariant defined as follows,

$$\hat{\mu} = T_1 \circ \mu = T_2 \circ f(x_i) = f(T_1 \circ \hat{x}_i). \tag{3.21}$$

To show the PNDE is rotational equivariant, the network is fully trained self-supervised on non-rotated and subsequently tested on rotated point cloud. A qualitative result is shown in Fig. 3.5, demonstrating the rotational equivariance in of density parameters by inferring a rotated point cloud, where the rotation imposed in the input point cloud is reflected in the output density parameters.

3.8 Chapter Summary

In this chapter, a novel Point cloud Neural Density Estimator (PNDE) derived from Gaussian mixture model is proposed to estimate the density of point cloud. The network adopts the concept of neural density estimation using maximum log-likelihood of multivariate Gaussian mixture model as an objective function for end-to-end network training. The output of PNDE is a set of density parameters and it can be directly used for classification and reconstruction tasks. The classification using estimated density parameters achieved comparable results to the state-of-the-art methods using $M = 64$ with overall accuracy of 93.67% in ModelNet10, 88.74% in ModelNet40 and 94.6% in ShapNetCore13. A network parameter compression strategy is implemented to compensate the use of independent networks in the Local Density Autoencoder (LDA). The size of PNDE can be further reduced up to $8\times$ smaller by adjusting the latent global density dimension size while retaining approximately 1.5% degradation in classification accuracy. Qualitative visualizations show that PNDE inherits the properties of permutation invariant and rotational equivariant toward input point clouds. Intuitively, the use of PNDE can be extended to perform point cloud reconstruction by adopting a decoder to form an autoencoder network. The concept is realized by viewing the estimated density as a rich and condensed representation of point cloud and subsequently sample output point cloud from the density. In the following chapter, the point cloud reconstruction network is proposed in light of PNDE for effective reconstruction output point cloud by generative sampling using a decoder.

Chapter 4

Partial Point Cloud Completion using Patch-based Autoencoder

4.1 Introduction

Partial point clouds is a common challenge in 3D acquisition of real-world objects mainly caused by viewpoint occlusion and limited sensors resolution [34, 35]. These limitations of 3D model acquisition have resulted in acquiring partial point cloud of the object from single viewpoint. The setting of multiple viewpoint acquisition sensors are commonly used to acquire a full 3D model of an object. Multiple viewpoint setup requires multiple cameras coordination and it requires the special controlling mechanism in performing data acquisition synchronization. A designated space has to be set up and it restricts the flexibility of an object scanning. Subsequently, a complex 3D reconstruction technique [25–27] is performed to merge all viewpoints into one complete model. As a consequence, it is not an ideal solution in applications that requires portability and real-time processing.

In recent research trend, point cloud reconstruction is commonly implemented using an autoencoder (AE) architecture which consists of a two-stage strategy [37]. The first stage is a point learning network (or it is known as an encoder) such as PointNet [4] and EdgeConv [52] to learn compressed latent representation of input point cloud. Learning compressed representation enables a deep learning network to learn a collection of shapes manifolds that is concise for task oriented deep learning networks. The procedure of learning compressed representation through directly processing raw point cloud is first realized in PointNet. In conjunction, PointNet solved the challenging nature of un-ordered and point cloud permutation by using point-wise convolution operation and symmetrical max-pooling function. As PointNet utilized point-wise convolutions, the learning lacks of correlation between neighbourhood points. Conversely, a graph-based approach [52] is introduced to extend the local receptive field to explore the relationship of neighbourhood points. It resulted in a better learning performance with richer compressed representation of point cloud from the local patches of point cloud.

Once the compressed latent representation is generated, a two-dimensional (2D) deformation learning network (or it is also known as decoder) such as FoldingNet [36] and AtlasNet [100] are widely implemented in the second stage. These decoders give an effective reproduction of output point cloud from the learnt compressed latent representation. Most of the existing methods [29, 149] make use of the properties of conventional convolutional network to generate a discrete data such as a typical linear interpolation method. However, these methods suffer from lower fidelity in reproducing a dense point cloud. To solve the problem, FoldingNet [36] proposed a novel 2D grid deformation learning method to generate point cloud through multi-stage folding procedures. In an incremental work, AtlasNet [100] introduced a patch-based point cloud decoding method by assembling multiple patches 2D deformed grid into a complete point cloud. As the result, AtlasNet is able to produce higher-fidelity output point cloud compared to FoldingNet.

Inspired from the AE architecture, a novel point cloud reconstruction technique is proposed in this paper using observable compressed representation (Gaussian components) for partial point cloud completion. A proposed two-stage architecture is demonstrated in Fig. 4.1, i.e. (i) Gaussian-based encoder and (ii) Patch-based decoder. The proposed Gaussian-based encoder learns to estimate the Gaussian components of partial point cloud. Each Gaussian component can be viewed as the governing basis function of each local patch. Subsequently, the patch-based decoder generates the output point cloud based on each estimated Gaussian components. Hence, a point cloud reconstruction network Gaussian point cloud autoencoder (GPAE) is proposed. The GPAE contains less learning parameters and less architectural complexity as compared to existing works such as AtlasNet [100], PCN [34] and MSN [35]. Moreover, by using Gaussian components as compressed representation, the GPAE showed a simplistic and tractable fashion in partial point cloud completion.

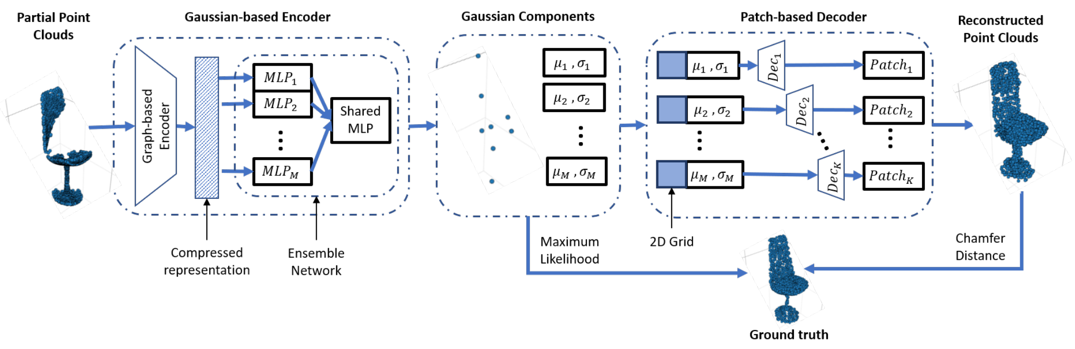


Figure 4.1: The proposed network,GPAE with two-stage network architecture: (i) Gaussian-based encoder, (ii) Patch-based decoder.

4.2 Gaussian Point Cloud Autoencoder

The GPAE has a two-stage architecture used for partial point cloud completion, where the two stages are: (i) Gaussian-based encoder and (ii) Patch-based decoder. The proposed Gaussian-based encoder is adopted from PNDE in Chapter 3 to learn to estimate the Gaussian components of complete point cloud by observing partial point cloud. Subsequently, the Patch-based decoder generates a collection of local patches using Gaussian components estimated from Gaussian-based encoder.

4.2.1 Stage 1: Gaussian-based Encoder (GE)

The GE denoted by $f_{GE} : \mathbb{R}^{3\times N_{par}} \mapsto \mathbb{R}^{6\times M}$ is a variant of a neural density estimator [54] tailored to estimate mixture of densities, i.e. Gaussian components. In this implementation, it infers M number of Gaussian components $\mathcal{Q}_m = [\mu_m, \Sigma_m]$ of complete point cloud by taking partial point cloud $S_{par} = \{p_{par,i} \in \mathbb{R}^3\}_{i=1}^{N_{par}}$ as input and a complete point cloud S_{gt} as ground truth.

$$\{\mathcal{Q}_m\}_{m=1}^{M} = f_{GE}(S_{par}). \tag{4.1}$$

Let $S_{gt} = \{p_{gt,i} \in \mathbb{R}^3\}_{i=1}^{N_{gt}}$ with N_{gt} number of point cloud in ground truth, where $N_{gt} > N_{par}$. By assuming S_{gt} exhibits M units of Gaussian components $\{\mathcal{Q}_m\}_{m=1}^{M}$, it can be defined such that $S_{gt} \sim \mathcal{N}(\{\mathcal{Q}_m\}_{m=1}^{M})$, where S_{gt} can be statistically sampled from the Gaussian components. To accomplish the Gaussian component inference, f_{GE} is comprised of a graph-based encoder adopted from [52] and an ensemble of M independent MLPs. In essence, the graph-based encoder encodes compressed representation from input partial point cloud with relationship of neighbouring points. Next, ensemble of independent MLPs infers individual Gaussian components from the encoded compressed representation of a complete point cloud. Intuitively, f_{GE} learns in prior to estimate $\{\mathcal{Q}_m\}_{m=1}^{M}$ by observing S_{par} and guided by S_{gt}. A maximum log-likelihood function in Eq. (4.2) is used as a loss function to guide f_{GE} ,

$$\mathcal{L}_{GE}(S_{gt}, \mathcal{Q}_m) = -\ln\left[\sum_{i=1}^{N_{gt}} \prod_{m=1}^{M} \mathcal{N}(p_{gt,i}|\mu_m, \Sigma_m)\right], \tag{4.2}$$

where,

$$\mathcal{N}(p_{gt,i}|\mu_m, \Sigma_m) = \frac{1}{(2\pi)^{\frac{3}{2}}|\Sigma_m|^{\frac{1}{2}}} e^{-\frac{1}{2}(p_{gt,i}-\mu_m)^T \Sigma_m^{-1}(p_{gt,i}-\mu_m)}. \tag{4.3}$$

In addition, by implementing a neural density estimator, procedures of inferring $\{\mathcal{Q}_M\}_{m=1}^{K}$ is turned into optimization problem by using gradient descent in back-propagating manner. Hence, it effectively mitigates the iterative terms in searching $\{\mathcal{Q}_m\}_{m=1}^{M}$ when relying on conventional methods such as k-means [150] and EM algorithm [151].

4.2.2 Stage 2: Patch-based Decoder (PD)

When image decoding is applied in 2D space, image pixels posses the grid and localization properties. In contrast, point cloud is un-grid and un-ordered [4]. Hence, an efficient approach in decoding 3D point cloud can be realized by 2D grid deformation. In recent work such as FoldingNet [36], a point-wise deconvolution layer is developed to learn 2D grid deformation for 3D point cloud sampling by inferring encoded latent feature. However, the drawback of FoldingNet sampling approach can be seen in lower fidelity sampling on local details. In addition, FoldingNet decoder takes iterations as it relies on multiple stages of deformation processes. On the other hand, AtlasNet [100] introduced an approach that samples multiple local patches and assembled them into full 3D point cloud. The method utilized one a scaled-down decoder per patch, hence vastly improved the decoding capability compared to FoldingNet decoder.

In GPAE, the PD denoted by $f_{PD} : \mathbb{R}^{6\times M} \mapsto \mathbb{R}^{3\times N_{out}}$ is adopted from AtlasNet, where it consists of K number of decoders,

$$f_{PD} = [f_{PD_1}, f_{PD_2}, ...f_{PD_M}], \tag{4.4}$$

and each decoder $f_{PD_m} : \mathbb{R}^{6\times 1} \mapsto \mathbb{R}^{3\times \frac{N_{out}}{M}}$ takes $(\mathcal{Q}_m, H_m)$ as input to decode local patch point cloud $S_{patch_m} \in \mathbb{R}^{3\times \frac{N_{out}}{M}}$ as follows,

$$S_{patch_m} = f_{PD_m}([\mathcal{Q}_m, H_m]), \tag{4.5}$$

where $H_m \in]0,1[^2$ is a random 2D grid. The notion of decoding a concatenated input $[\mathcal{Q}_m, H_m] \in \mathbb{R}^{(6+2)\times \frac{N_{out}}{M}}$ indicates the 2D grid deformation guided by the Gaussian components $[\mu_m, \Sigma_m]$ where μ_m acts as the centroid of S_{patch_m} and Σ_k is the boundary of sampling region. Therefore, the decoding process can be viewed as a local patch sampling process $S_{patch_m} \sim \mathcal{N}([\mathcal{Q}_m, H_m])$. Subsequently, the output point cloud $S_{out} \in \mathbb{R}^{3\times \mathcal{N}}$ is defined as a collection of local patches S_{patch_m} and can be viewed as $S_{out} = \{S_{patch_1}, S_{patch_2}, ...S_{patch_M}\}$.

Two common 3D metrics, namely Chamfer distance (CD) and Earth Mover Distance (EMD) are implemented as loss function to optimize a point cloud decoder [34, 35]. In this work, two variants of GPAEs are trained using CD and EMD as reconstruction loss $\mathcal{L}_{PD}$. The CD from [109] denoted as d_{CD} is defined as,

$$d_{CD} = \frac{1}{|S_{gt}|} \sum_{p_{gt}\in S_{gt}} \min_{p_{out}\in S_{out}} ||p_{gt} - p_{out}||_2^2 + \frac{1}{|S_{out}|} \sum_{p_{out}\in S_{out}} \min_{p_{gt}\in S_{gt}} ||p_{gt} - p_{out}||_2^2, \tag{4.6}$$

where p_{gt} and p_{out} are the point cloud from the ground truth S_{gt} and reconstructed output S_{out}. The measured similarities are the L_2 norm distance of two nearest point that is computation efficient and permutation invariant.

On the other hand, EMD is a bijective similarity metric that matches the local criterion such as point cloud density, denoted as d_{EMD} and is defined as,

$$d_{EMD}(S_{out}, S_{gt}) = \min_{\phi: S_{out}\rightarrow S_{gt}} \sum_{p_{out}\in S_{out}} ||p_{out} - \phi(p_{out})||_2, \tag{4.7}$$

where ϕ is a bijection that maps matching S_{out} and S_{gt}. As EMD has $O(n^2)$ complexity, CD is often chosen for its computation efficiency. There are several $O(n)$ approximations of EMD such as [35, 110]. In the implementation of this work, EMD from [35] is adopted for comparisons to existing works. Therefore, the loss functions of the proposed GPAE is defined as follows,

$$\mathcal{L}_{GPAE} = \mathcal{L}_{GE} + \mathcal{L}_{PD}, \tag{4.8}$$

where $\mathcal{L}_{PD} = d_{CD}$ for CD training variant of GPAE and $\mathcal{L}_{PD} = d_{EMD}$ for EMD training variant of GPAE.

4.3 Experimental Setup

Two experiments are conducted to perform ablation study of the proposed GPAE. A description of dataset, training setup and network configuration are given in detail in the following subsections.

4.3.1 Dataset and training setup

The performance of the GPAE network are evaluated using the ShapeNetCore dataset [145] containing 55 common object categories with about 51,300 unique 3D models. Similar pre-processing settings are adopted from [35] that extract 8 synthetic CAD models categories with 30,974 unique 3D models of table, chair, car, airplane, sofa, lamp, vessel, and cabinet, in which 24,779 and 6,195 3D models are split as training and testing samples. In the setting, all ground truth samples are uniformly sampled with $N_{gt} = 8192$ and partial samples are sampled on each 3D model in 50 randomly selected viewpoints with $N_{par} = 5000$. The total number of training and testing partial samples are 1,238,950 and 309,750.

In network training detail, the GPAEs are limited to only 16 viewpoints randomly selected from the total pool of training partial samples. Besides that, probabilistic sampling learning is enforced in the training of GPAEs, where each partial sample is further downsampled from $N_{par} = 5000$ to $N_{par} = 2048$ and the relative ground truth

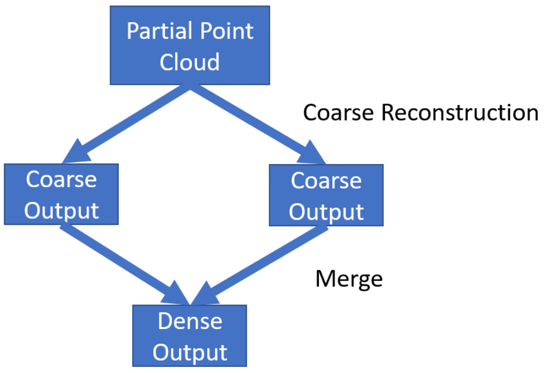


Figure 4.2: The process of producing dense output using GPAEs.

sample is downsampled from $N_{gt} = 8192$ to $N_{gt} = 4096$. In the network testing, 50 viewpoints samples are used with similar $N_{par} = 2048$ and $N_{gt} = 4096$ downsampling on partial samples and ground truth samples. To obtain dense output point cloud during inference, the GPAE is two times forwarded on two sets of partial point cloud to generate output point cloud with $N_{out} = 2 \times 4,096 = 8,192$. Therefore, a coarse output of $4,096$ point cloud is resulted from single forward propagation and a dense output of $8,192$ point cloud is resulted from two-times forward propagation as shown in Figure 4.2.

4.3.2 Network configuration

The configuration of GPAE network is standardized for all experiments as shown in Table 4.1. For the first stage, the graph-based encoder from [52] is adopted with $3-64-128-128-256-d_{in}$ network configuration in shared MLPs and k nearest neighbour of $k = 10$. It takes partial point cloud with dimension of $\mathbb{R}^{3\times 2048}$ as an input and generates an output of a compressed representation with dimension of $\mathbb{R}^{1\times d_{in}}$. Subsequently, an ensemble network consisting of M independent MLPs with configuration $\{d_{in}-d_{out}\}_M$ in parallel and an aggregating shared MLP $\{d_{out}-256-6\}$ are implemented for individual Gaussian component inference. The dimension of aggregated Gaussian components is viewed as $\mathbb{R}^{6\times M}$. Rectified Linear Unit activation functions (ReLU) and $1D$ batch

Table 4.1: Configuration of GE and PD. In the implementation of GPAE, $K = M$ is used to match the number of independent decoder and number of Gaussian components.

Layers	Configuration	Output Dim.
Input	-	$N \times 3$
GE	EdgeConv ($k = 10$) $[3, 64, 128, 128]$	$N \times 128$
	Conv1D $[128, 256]$ +BN+ReLU	$N \times 256$
	Conv1D $[256, d_{in}]$ +BN+ReLU	$N \times d_{in}$
	Maxpool1D (N)	$1 \times d_{in}$
	Superposed MLP $(d_{in} \times d_{out}) + (d_{in} \times M) + (d_{out} \times M)$ +BN+ReLU	$M \times d_{out}$
	Conv1D $[d_{out}, 256]$ +BN+ReLU	$M \times 256$
	Conv1D $[256, 6]$ +BN	$M \times 6$
Gaussian Components	ReLU ($M \times 3$)	$M \times 6$
	Softplus ($M \times 3$)	
Patches	Expand($\frac{N_{out}}{K}$)	$M \times \frac{N_{out}}{K} \times 6$
PD	$K\times$ Shared MLP $[(6+2), 1024]$ +BN+ReLU	$K \times \frac{N_{out}}{K} \times 1024$
	$K\times$ Shared MLP $[1024, 512]$ +BN+ReLU	$K \times \frac{N_{out}}{K} \times 512$
	$K\times$ Shared MLP $[512, 256]$ +BN+ReLU	$K \times \frac{N_{out}}{K} \times 256$
	$K\times$ Shared MLP $[256, 3]$ +BN+Tanh	$K \times \frac{N_{out}}{K} \times 3$
-	Concatenation	$N_{out} \times 3$

normalization (BatchNorm1D) are used throughout the entire network with an exception on the last layer of the aggregating shared MLP, where linear and softplus activation functions are applied on each $\mathbb{R}^{3\times M}$ dimension of the aggregated Gaussian components.

For the second stage, K independent decoder is adopted from [100] with shared MLPs configured as $\{(6+2)-1024-512-256-3\}_K$, where $(6+2)$ indicates the concatenation of each Gaussian component and a random 2D grid $[\mathcal{Q}_m, H_m]$. It takes the aggregated Gaussian components from first stage and output K patches of reconstructed point cloud with dimension of $\mathbb{R}^{3\times\frac{N_{out}}{K}\times K}$. By concatenating the patches, a full reconstructed point cloud is obtained with dimension $\mathbb{R}^{3\times N_{out}}$. ReLU and BatchNorm1$D$ are implemented throughout the decoder with exception of hyperbolic tangent activation function on the last layer of K decoder.

To efficiently deploy the independent networks in GPAE, i.e. M independent MLPs and K decoders, a parameters compression strategy [143, 152] is implemented to apply the concept of superposition for weights decomposition as shown in Table **??**, where d_{in} and d_{out} are the input and output dimension of MLP. By factoring the independent network forward process into matrix vector, the network is more optimized for GPU parallel computation with less parameters. For an instance, let $d_{in} = 1024$, $d_{out} = 512$ and $M = 8$, 87.2% of parameters from independent configuration can be reduced with negligible performance degradation.

In the experiment evaluation, the number of Gaussian components is set $M = 8, 16$ and is denoted by GPAE-8 and GPAE-16 respectively. The number of patches in PD is set $K = M$ for both GPAE-8 and GPAE-16. To further optimize GPAE model, the network is developed on mixed precision where all trainable parameters are set using half precision and batch normalization parameters are remained as full precision. All experiments are executed for 50 epoch with learning rate of 1×10^{-3} and scheduled decay rate of 0.5 per 10 epoch. Parameters are initialized using Xavier normal and batch size 256 is used. ADAM optimizer is implemented as optimization method for the network training. The specifications of test bench for the experimentation are 2× RTX TITAN with Pytorch 1.8, CUDA 11.1 and Distributed Data Parallel learning enabled.

4.3.3 Evaluation metrics

In performance evaluation, CD in Eq. (4.6) and EMD in Eq. (4.7) are used to evaluate GPAE and existing works. Comparison to existing methods: AtlasNet [100], PCN [34], MSN [35] are using 8,192 dense point cloud, while GRNet [38] is using 16,382 dense point cloud. In addition to CD and EMD, F-Score in [38] is used to measure a more concise point cloud reconstruction performance defined as follows,

$$F_{score}(s) = \frac{2P(s)R(s)}{P(s) + R(s)}, \tag{4.9}$$

where P(s) and R(s) denote the precision and recall for a distance threshold s respectively.

$$P(s) = \frac{1}{nS_{out}} \sum_{p_{out} \in S_{out}} \left[\min_{p_{gt} \in S_{gt}} ||p_{gt} - p_{out}|| < s \right] \tag{4.10}$$

$$R(s) = \frac{1}{nS_{gt}} \sum_{p_{gt} \in S_{gt}} \left[\min_{p_{out} \in S_{out}} ||p_{out} - p_{gt}|| < s \right], \tag{4.11}$$

4.4 Experiment #1: Performance Evaluation on GPAEs

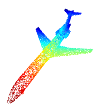
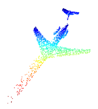
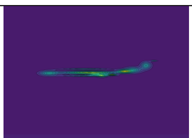
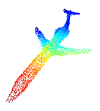
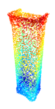
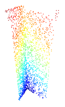
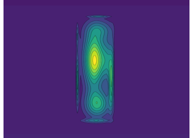
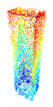
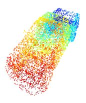
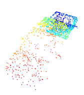
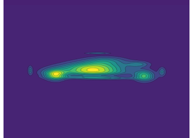
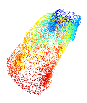
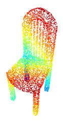
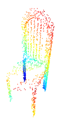
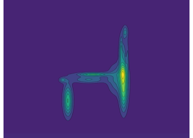
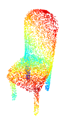
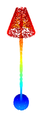
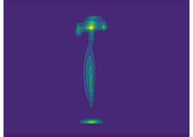
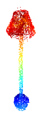
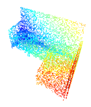
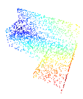
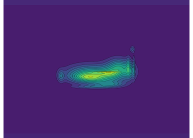
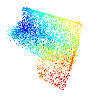
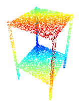
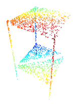
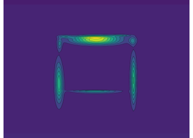
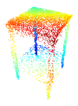
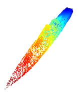
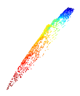
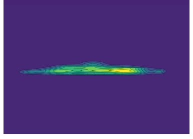
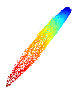
Two variants of GPAE, i.e. GPAE-8 and GPAE-16 is used to perform some ablation analysis to understand its effect in point cloud reconstruction. Several examples such as airplane, vessel and chair are visualized from GPAE-8 in Table 4.3 for the demonstration of the process of point cloud reconstruction. In the visualization, the stages of completion from the input partial point cloud, to the estimated Gaussian components which contain the means and variances and to the reconstructed point cloud are shown. The Gaussian components are depicted as a heatmap by computing the average sum of variances.

Table 4.2 shows the performance of CD and EMD variant of GPAE, evaluated using CD and top 1% in F-Score (F@1%) evaluation metric on coarse $4,096$ and dense $8,192$ output point cloud. From the CD measurements, a drastic reconstruction improvement is observed on dense point cloud over coarse point cloud for all variants of

Table 4.2: Measured average CD with $N_{out} = 8,192$ of GPAE trained using CD and EMD on K=8 and K=16 patches.

Variant	Output / Models	Coarse		Dense	
		CD	F@1%	CD	F@1%
CD	GPAE-8	0.73	0.39	0.58	0.52
	GPAE-16	0.70	0.40	0.56	0.53
EMD	GPAE-8	0.87	0.40	0.74	0.53
	GPAE-16	0.83	0.42	0.70	0.54

Table 4.3: Results of reconstructed partial point cloud completion using GPAE with input partial point cloud $N_{par} = 2,048$ and dense output point cloud $N_{out} = 8,192$. The Gaussian components $\{\mathcal{Q}_m\}_{m=1}^{M}$, where $M = 16$ are visualized as a heatmap plot.

	Ground Truth	**Partial PC**	**Gaussian components**	**Recon-structed PC**
Airplane				
Cabinet				
Car				
Chair				
Lamp				
Sofa				
Table				
Vessel				

GPAE. However, EMD variant of GPAE showed higher CD loss compared to CD variant. Although, EMD attempts to match local criterion such as density for more discriminative reconstruction, CD tends to compute the mean of distance of closest points, in which the local criterion is neglected [35]. As suggested in [38, 153], F-Score in Eq. (4.9) is used to better measure the reconstruction accuracy of the GPAEs. Thus, in terms of F@1%, EMD variant of GPAE performed better than CD variant by a slight margin. Nonetheless, the adopted EMD has far higher compute complexity compared to

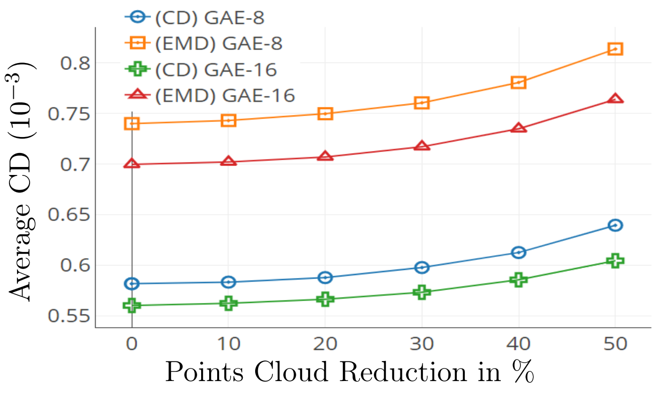


Figure 4.3: Loss evaluation of GPAE-8 and GPAE-16 with the number of point cloud reduction from partial input tested on the models.

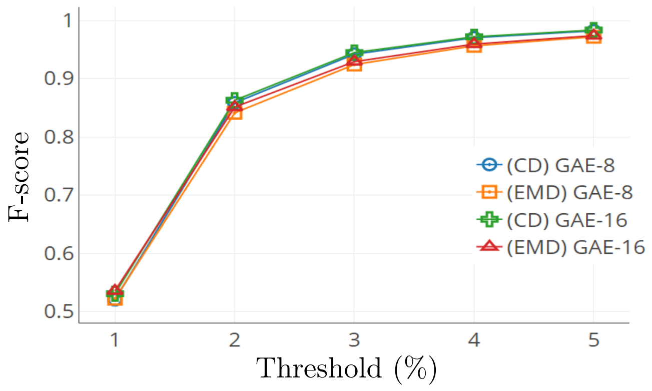


Figure 4.4: F-score evaluation of GPAE-8 and GPAE-16 on the variation of distance threshold.

CD. Furthermore, the GPAE-16 showed lower CD reconstruction loss and higher F@1%. This is due to increased number of Gaussian components which is equivalent to increased number of basis function that can capture the local structure of point cloud. Overall, a slight improvement in CD loss and F-score is achieved when the number of Gaussian components is increased from $M = 8$ to $M = 16$.

In Fig. 4.3 and Fig. 4.4, the evaluation is performed on dense $8,192$ output S_{out}. Fig. 4.3 reports the effects on CD and EMD variant of GPAE-8 and GPAE-16 towards sparse S_{par} with missing points. To obtain sparse S_{par}, the point cloud is downsampled from 0-50% and re-upsampled to N_{par}. From the average CD plot, an average of 8% reconstruction degradation at 50% missing points in S_{par}, suggesting the degree of robustness towards sparse input partial point cloud. While, Fig. 4.4 reports the F-Score against top 1-5% threshold imposed on the precision and recall between the

S_{out} and S_{gt}. All GPAE models achieved average of F@1% $\sim$ 0.53, F@2% $>$ 0.85 and F@5% $>$ 0.97. This indicates a well within 5% deviation in reconstruction accuracy from the GPAE models against the ground truth.

4.5 Experiment #2: Performance Evaluation with Benchmark Methods

In benchmark performance comparisons, the average CD of coarse and dense partial point cloud completion is evaluated in Table 4.4 as compared to AtlasNet [100], PCN [34] and MSN [35]. All models are trained with identically pre-processed dataset and output of dense 8,192 output point cloud. Both GPAE-8 and GPAE-16 outperformed all other works, but only with incremental improvement from GPAE-8 to GPAE-16. Although GPAE-16 showed incremental improvement over GPAE-8, it indicates that the density of Gaussian components on GPAE-8 can be potentially further saturated using denser ground truth point cloud. By adopting concept of network weights superposition, the number of parameter of GPAEs is drastically reduced and is relatively lower compared other models. In addition, other works apply a single latent compressed representation as the basis function to govern a set of complete point cloud. Hence, the $K = 16$ decoders in AtlasNet and MSN have a shared latent compressed representation for point cloud generation. Conversely, the proposed GPAE-8 and GPAE-16 use $K = 8, 16$ decoders, where each decoder uses an independent Gaussian component as the governing basis function for each local point cloud sampling. This operation offers an advantage on each decoder to avoid ambiguity and redundancy when generating point cloud. As stated in [104, 154], a set of complete point cloud is known to be governed by

Table 4.4: Average CD with $N_{out} = 8,192$, number of trainable parameters and number of patches K compared to existing methods.

Methods	Average CD ($\times 10^{-3}$)		Refine	#param ($\times 10^{6}$)	K
	Coarse	Dense			
AtlasNet	-	1.82	-	9.73	16
PCN	-	1.21	✓	6.58	-
MSN	1.29	1.00	✓	10.59	16
GPAE-8	0.73	0.58	-	**1.51**	8
GPAE-16	**0.70**	**0.56**	-	1.55	16

Table 4.5: Quantitative comparison on per-class CD and average CD between the proposed method and existing methods. The multiply ratio is 10^{-3} and $N_{out} = 8,192$.

Methods	Airplane	Car	Chair	Lamp	Sofa	Cabinet	Vessel	Table	Average CD
AtlasNet	0.85	1.42	1.58	1.82	2.67	2.49	2.30	1.46	1.82
PCN	0.66	1.10	1.41	1.46	1.36	1.23	1.36	1.14	1.21
MSN	0.56	1.03	1.02	1.07	1.16	1.19	0.99	0.96	1.00
GPAE-8	**0.53**	**0.51**	**0.57**	0.79	0.58	**0.56**	**0.47**	0.60	0.58
GPAE-16	0.61	0.54	0.59	**0.52**	**0.57**	0.57	0.53	**0.53**	**0.56**

Table 4.6: Quantitative comparison on per-class EMD and average EMD between the proposed method and existing methods. The multiplier of the result is 10^{-3} and $N_{out} = 8,192$.

Methods	Airplane	Car	Chair	Lamp	Sofa	Cabinet	Vessel	Table	Average EMD
AtlasNet	3.27	4.20	5.03	10.71	6.97	8.91	8.11	5.07	6.53
PCN	3.44	4.44	6.89	15.45	6.28	8.79	6.56	6.84	7.34
MSN	**2.18**	**3.28**	3.63	6.04	**3.47**	4.16	**3.83**	**3.66**	**3.78**
GPAE-8	4.00	3.32	**3.14**	4.09	3.90	**3.96**	4.23	4.00	3.96
GPAE-16	3.73	3.82	3.89	**4.02**	3.88	4.04	4.15	3.98	3.94

several basis functions, therefore it is insufficient to capture much shape information of a 3D model using single latent compressed representation. Some models such as PCN and MSN inserted an extra step of refinement to generate dense point cloud, however, it leads to higher model complexity and increased number of model parameters.

To have a better understanding in the generalization of objects partial point cloud completion, the per class average CD and per class average EMD are tabulated in Table 4.5 and Table 4.6. Notably, GPAE-8 showed the difficulty in reconstruction of "lamp" class as it was also posited in [35], where the "lamp" class is isolated compared to other classes with more samples. However, GPAE-16 showed its ability in encountering such a case of isolated class. As GPAE-16 has more Gaussian components that govern a local point cloud feature, it is able to learn more shapes information by learning the density of point cloud distribution. On the other hand, MSN outperforms all models on average EMD. This can be resulted by the refinement process found in MSN where the input partial point cloud is merged into output point cloud, hence leading to higher similarity in local criterion that gives a lower EMD. As opposed to CD, it measures similarity based on nearest points. Nonetheless, both GPAE-8 and GPAE-16 have a more uniformed EMD across per class average EMD compared to MSN, suggesting a more generalized shape distribution learning capability. Moreover, GPAE-8 and GPAE-

Table 4.7: Evaluation of average CD and F-score@1% on dense $N_{out} = 16,384$ point cloud using Shapenet dataset.

Methods	Refine	Average CD ($\times 10^{-3}$)		F@1%		#param
		Coarse	Dense	Coarse	Dense	($\times 10^6$)
GRNet	-	1.13	0.45	0.34	0.62	69.43
GRNet	✓	0.93	**0.27**	0.39	**0.71**	76.70
GPAE-8	-	0.73	0.52	0.39	0.58	**1.51**
GPAE-16	-	**0.70**	0.50	**0.41**	0.60	1.55

16 can reconstruct better shape in "chair, lamp, cabinet" classes over MSN.

In addition, it is also noteworthy in AtlasNet, the dense output point cloud is achieved by combining outputs of four passes on four different inputs [35]. In models such as MSN and PCN, they generate dense point cloud in a coarse to fine fashion. The models first generate a coarse point cloud then deploy a points residual network to generate dense point cloud from the coarse point cloud. In the similar manner, dense outputs of the proposed GPAEs are combination of two passes of two randomly sampled on the same input and are generated from its Gaussian components. Despite that, the proposed GPAEs outperformed other methods by a great margin. This is due to the proposed GPAE using density distributions (Gaussian components) as local feature representation that represents the local distribution of point cloud respectively. Naturally, a Gaussian component establishes its probabilistic distribution (centroid, i.e. mean, and boundary, i.e. variance) by observing priors (S_{gt}). By leveraging probabilistic sampling on training and testing samples and enough network training iterations, the posterior density of GPAEs can be established when all S_{gt} data points are observed. As the results from probabilistic sampling learning, the generation of dense point cloud is at a cost of a smaller network compared to other networks that direct fully generate dense point cloud.

Another performance comparison between the latest work on GRNet [38] and the proposed GPAE is shown in Table 4.7. In the comparison, two refinement variants of GRNet are put into comparison and the dense output point cloud generated by GRNet is 16,384. The GPAEs could generate similar 16,384 dense output point cloud by combining four times forward passes of reconstructed point cloud. The average CD of coarse output of GPAE outperformed both refinement variant of GRNet in terms of average CD and F@1%. However, the GPAEs show a significant gap on dense output compared to

GRNet with refinement setting turned on. This suggests a refinement process for GPAE can close the gap on partial point cloud completion performance comparing to GRNet. Despite the better performance in GRNet, GPAEs have approximately 50 folds smaller network size as compared to GRNet. Therefore, the use of the proposed GPAE model in real-time inference on multiple 3D object instances is more practical in producing dense output from partial point cloud.

4.6 Experiment #3: Performance Evaluation on KITTI Dataset

To further evaluate the point clouds completion, KITTI dataset [155] containing real-world point cloud captured from LiDAR scanning is used to perform the benchmark comparison between the proposed GPAE-16 method with the existing works. Data samples are pre-annotated and readily labeled. The partial point clouds in KITTI dataset are highly sparse and do not have complete point clouds as ground truth. The setup of this evaluation is performed according to [38] by using two performance metrics, i.e. Consistency [34] and Uniformity [156]. Consistency is defined as the average CD between two consecutive frames t_i and t_{i-1} of j-th reconstructed car instance S^j_{out} over number of frames n_f. It is measured as follows [34],

$$\text{Consistency} = \frac{1}{n_f - 1}\sum_{i=2}^{n_f} d_{CD}(S^j_{out,t_{i-1}}, S^j_{out,t_i}), \tag{4.12}$$

Uniformity is the measure of distribution uniformity of the reconstructed point cloud by measuring the local and non-local distribution uniformity defined as follows [156],

$$\text{Uniformity}(p) = \frac{1}{M}\sum_{i=1}^{M} \mathrm{U}_{\text{imb}}(S_{out,i}) \cdot \mathrm{U}_{\text{clut}}(S_{out,i}), \tag{4.13}$$

where $S_{out,i}$, $i = 1, \ldots, M$ is a subset of point cloud cropped from S_{out} using farthest point sampling and ball query of radius $r = \sqrt{p}$ and p is the percentage of points in $S_{out,i}$. The term U_{imb} is accounted for non-local uniformity is defined as,

$$\mathrm{U}_{\text{imb}}(S_{out,i}) = \frac{(|S_{out,i}| - \hat{n})^2}{\hat{n}}, \tag{4.14}$$

Table 4.8: Evaluation of Consistency and Uniformity with percentage of points p on KITTI dataset [155].

Methods	Consistency ($\times 10^{-3}$)	Uniformity ($\times 10^{-3}$)				
		$p = 0.4\%$	$p = 0.6\%$	$p = 0.8\%$	$p = 1.0\%$	$p = 1.2\%$
AtlasNet	0.700	1.146	1.005	0.874	0.761	0.686
PCN	1.557	3.662	5.812	7.710	9.331	10.823
MSN	1.951	0.822	0.675	0.523	0.462	0.383
GRNet	**0.313**	0.632	**0.572**	**0.489**	**0.410**	**0.352**
GPAE-16	1.446	**0.438**	1.024	1.869	2.887	3.859

where $\hat{n} = rNp$. Subsequently, the term $\mathrm{U}_{\mathrm{clut}}$ is accounted for local uniformity defined as,

$$\mathrm{U}_{\mathrm{clut}}(S_{out,i}) = \sum_{k=1}^{|S_{out,i}|} \frac{(d_{i,k} - \hat{d})^2}{\hat{d}}, \tag{4.15}$$

where $d_{out,i,k}$ is the k-th nearest point in $S_{out,i}$ and $\hat{d}$ is the expected point-to-neighbour distance.

Table 4.8 shows the Consistency and Uniformity comparison results on GPAE-16 and existing works. The GPAE-16 achieved better consistency over PCN and MSN, however it falls short of AtlasNet and a larger gap compared to GRNet. In addition, GPAE-16 achieved better uniformity with $p = 0.4\%$ compared to all exiting works, however degradation of uniformity is observed with increased $p = 0.6\%, 0.8\%, 1.0\%, 1.2\%$ and is outperformed by existing works, except for PCN.

One of the factors that caused larger gap of consistency between AtlasNet and GRNet and the declination of uniformity is due to all existing works which are fine-tuned on ShapeNetCars [38] where the dataset only contains single class object, i.e. cars. In contrary, PCN and GPAE-16 are trained on ShapeNet which contain multiple classes of object. Moreover, GPAE-16 is a non-residual reconstruction model compared to most residual completion model, i.e. PCN, MSN, and GRNet, that has higher capability in reconstructing higher fidelity and consistent dense point cloud. Figure 4.5 demonstrates several partial point cloud completion of cars from KITTI dataset. From the demonstration, although GPAE-16 is optimized on regular ShapeNet it is able to reconstruct reasonable outputs from sparse real-world point cloud. However, the GPAE-16 is not able to concisely reconstruct the output in the instances such as Figure 4.5(c) and 4.5(d), where input details are excessively absent.

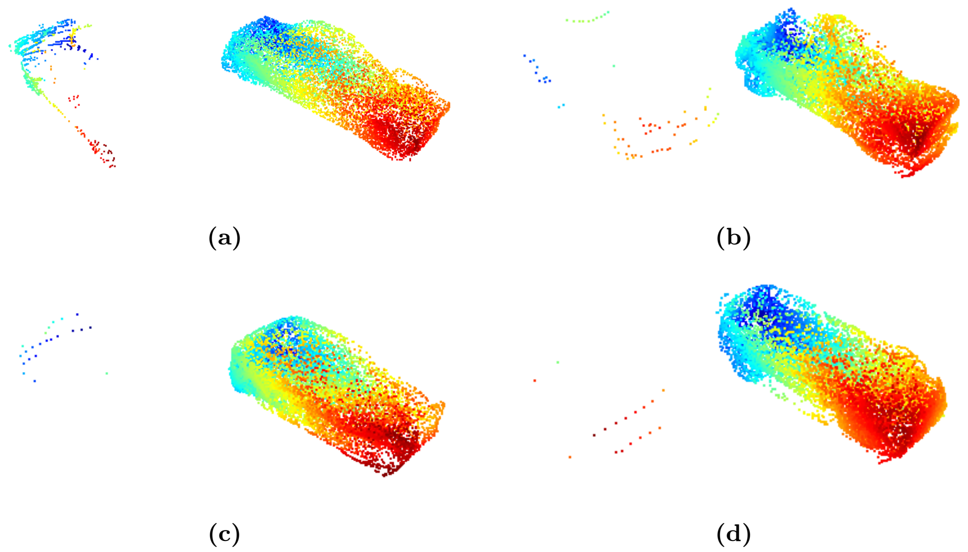


Figure 4.5: Four sample results of partial point cloud (left) completion on cars and completed point cloud (right) from KITTI dataset at random timestamps.

4.7 Chapter Summary

A novel partial point cloud completion model using Gaussian component as compressed representation, GPAE is proposed in this chapter to reconstruct a complete 3D points from single viewpoint partial point cloud. The GPAE-8 and GPAE-16 outperformed all other models that similarly generate 8,192 dense output point cloud with lowest average CD of 0.56×10^{-3}. Although the GPAEs do not possess point cloud refinement process, they showed comparative performance in EMD comparison. Moreover, the GPAEs have approximately 10 folds smaller network size compared to models that generate 8,192 dense output and 50 folds to models that generate 16,384 dense output. The GPAEs also showed robustness toward sparse input partial point cloud by facing an average of 8% reconstruction degradation at 50% missing points. While, 97% reconstruction accuracy is achieved within 5% deviation relative to ground truth. Overall, the GPAEs showed generalized point cloud shapes learning indicated by the uniform CD and EMD across 3D objects. Although GPAE-16 is non-residual reconstruction model, it achieved reasonable results in the evaluation of real-world point cloud. In the following chapter, the proposed GPAE is extended to learn point cloud that is imposed on transformation, i.e. rotation using a novel network architecture in this research field.

Chapter 5

Part-to-Whole Learning Capsule Network on Point Cloud Classification and Reconstruction

5.1 Introduction

Point cloud objects recognition is crucial in the recent advancement of autonomous robotic applications [157] and semantic environment learning from depth sensors [20, 158]. The use of artificial neural networks (ANN) to learn semantic 3D learning such as object detection, object classification, and 3D mapping are benefited from data-driven learning on depth perception from real-world environment. Several popular 3D data representation are depth-map (RGB-D) and point clouds, where RGB-D is a data format of a RGB image coupled with a depth-map and point cloud is a sparse data point sampled from an object surface [159]. Comparatively, representation of 3D object in point cloud could preserve richer spatial and geometric information than RGB-D due to its sparsity and non-grid structure. However, the process of dealing with point clouds efficiently possesses a set of challenges, such as points permutation and sparsity of data points [4]. In general, three learning models are used to learn and encode 3D point clouds for classification, i.e. point-based model, graph-based model, and capsule-based model.

Point-based models [34, 160] are the early works that implemented deep learning network for efficient point clouds learning. PointNet [4] is one of the pioneering work that devised a shared MLP-based network to directly process raw input point clouds. The network is modeled based on Hausdorff distance and universal approximation theorem. In addition, with implementation of max pooling in PointNet, the network is permutation invariance toward input point clouds. In its incremental work, PointNet++ [7] implemented recursive PointNet to achieve local feature grouping using farthest point sampling as guidance. In converse to PointNet, where it only encodes global features, PointNet++ aggregates the local features relative to the encoded global features. Therefore, PointNet++ is able to learn better than PointNet.

As point-based models are more emphasized on point-wise learning and addition steps are required to aggregate the local features, graph-based models [95, 96, 161] learn the set of point clouds by drawing a directed graph of each point, i.e. vertices and edges. Benefited from directed graph, such model is able to aggregate local information directly in a sparse data environment. In addition, by aggregating hierarchical of directed graphs, a coarse graph can be produced to group the local features as a global feature. In one of the early graph-based models, Valsesia et al. [95] proposed a graph neural network to synthesize object shapes. An adjacency matrix is calculated by using the directed graph features of each vertex in each graph convolution layer. Despite its superior results, the calculation of the adjacency matrix requires quadratic computation complexity and consumes a lot of memory. In DGCNN [52], a graph is dynamically constructed on each local point cloud and projected into high dimensional feature space for network learning. Multiple variants of DGCNN [162–164] are devised to improve its performance while reducing the model size.

In contrary to existing works that learn by extracting most significant features and aggregate them from local to global level, capsule network is proposed to enforce part-to-whole learning of an object. For instance, capsule network in [41] implemented equally strided convolution layers to extract local part features of an object in 2D images. Following that, Dynamic Routing (DR) algorithm as a soft clustering algorithm is devised to cluster the extracted part features into an object capsules, where each object capsule contains object specific features. The latter, Hinton et al. [70] proposed Expectation-Maximization (EM) routing algorithm to enhance the soft clustering step. Despite initial implementations of capsule network are on 2D images, Zhao et al. [104] proposed a capsule network to learn point cloud objects. The method implemented replicas capsule encoders and DR to obtain latent capsules that represent the point cloud object.

Nonetheless, most aforementioned techniques in point clouds feature learning mainly perform best when input point clouds are ideally conditioned, i.e. canonicalized point clouds [4]. This is due to the networks that are not generalized towards variation in the input point clouds [165], such as rotational transformation, which is common in practical applications. Ultimately, rotation equivariance is a desired property in a network that allows learning of transformation invariant features, while retaining the original transformation that is imposed in the input samples [146]. To encounter such variation, Spatial Transformer Network (STN) is proposed to canonicalise the augmented input point clouds before a learning process [4, 52]. However, due to limitation of STN it fell short to a small tolerance of rotation degree. In other approaches [94, 166], local reference frames based on points normal are estimated in the input samples to establish global rotation invariance. Besides establishing invariance in the input sample, invariance in the latent feature is established in the networks [167, 168]. In recent capsule network [42], rotation perturbation on input point clouds during network training is employed to enforce the rotation equivariance. Several discrete viewpoints are utilized to draw agreement on object existence using DR. Further, capsule network in [48] generated numbers of local reference frames as local features followed by DR to achieve rotation equivariance in point clouds learning.

Inspired by part-to-whole relationship learning and rotation equivariance in capsule networks [42, 48, 104], a part-based capsule network devised from a Part Sampler Network (PSN) and capsule network is proposed for point clouds learning. To motivate and demonstrate the benefit from enforced part-to-whole learning of a capsule network, the proposed network utilizes locally segmented parts from input point clouds for parts reasoning to learn point cloud object. The reasoning mechanism is driven by the Dynamic Routing that allows voting on existence of point cloud object by parts. This is mainly achieved by searching for maximum response of a part feature corresponding to object class capsule using operation such as dot product. In order to consistently segment parts from input point clouds, PSN is developed based on Point cloud Neural Density Estimator (PNDE) in Chapter 3 to estimate the density parameters of input point clouds. Using the estimated density parameters, local parts of input point clouds are segmented based on the likelihood of points. Moreover, PSN is rotation equivariance due to the PNDE backbone. In addition, the property rotation equivariance can be inherited by adopting capsule network architecture [48, 146]. Hence, the proposed network is effectively rotation equivariance. Further, a contrastive learning method is implemented to enhance the network training process through a redundancy reduction technique [169].

5.2 Part-based Capsule Network

The proposed network shown in Figure 5.1 comprises two stages: (i) Parts segmentation on the input point cloud using PSN, and (ii) Part-to-whole relationship learning using capsule network with Dynamic Routing algorithm. To ensure rotation equivariance of the proposed network, several required property are defined as follows,

Definition #1 (Rotational Equivariant Network) For a network to be rotational equivariant, all function map is required to be an equivariant map and the resultant rotation in the input causes equivalently transformation in the output [146, 147], such that:

$$\Lambda(g_1 \circ \mathcal{X}) = g_2 \circ \Lambda(\mathcal{X}), \tag{5.1}$$

where $\{g_1, g_2\} \in \mathcal{G}$ and $\mathcal{G}$ is an affine transformation group. Λ is an equivariant function map and $\mathcal{X}$ is data group.

Definition 2 (Rotational Equivariant Local Maximizer) A density estimator is equivariant under linear affine transformation, i.e. rotation. Given a random variable $\mathcal{X}$ is sampled from μ and Σ:

$$\mathcal{X} \sim \mathcal{N}(\mu, \Sigma). \tag{5.2}$$

Let $\hat{\mathcal{X}} = A\mathcal{X} + b$ be an affine transformation acted on $\mathcal{X}$, then $\hat{\mathcal{X}}$ can be sampled from

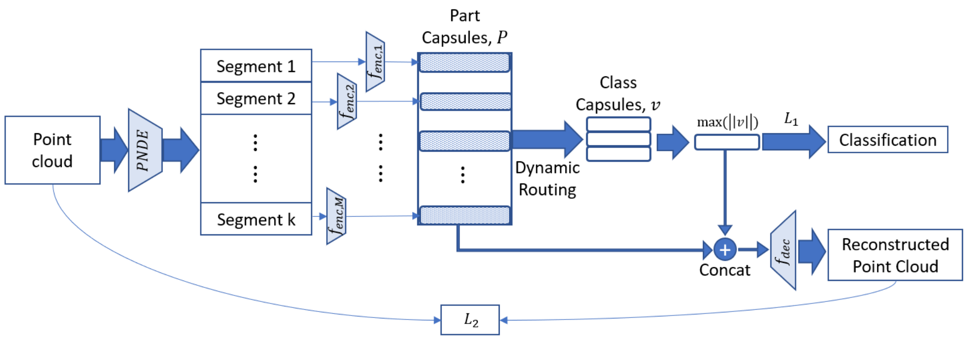


Figure 5.1: The proposed point clouds objects classification model using PNDE and a capsule network, optimized using a joint loss consisted margin loss, L_1 and Chamfer distance, L_2.

density parameters of $\hat{\mu}$ and $\hat{\Sigma}$:

$$\hat{\mathcal{X}} \sim \mathcal{N}(\hat{\mu}, \hat{\Sigma}). \tag{5.3}$$

where $\hat{\mu} = A\mu + b$ and $\hat{\Sigma} = A\Sigma A^T$. As stated in [148], let $\mathcal{N}(\mu, \Sigma|\mathcal{X})$ be the local maximizer for $\mathcal{X}$, then $\mathcal{N}(\hat{\mu}, \hat{\Sigma}|\hat{\mathcal{X}})$ is inherently the local maximizer for $\hat{\mathcal{X}}$.

Definition 3 (Equivariant Weighted Mean Operation) Let a weighted mean operation $\mathcal{F} : \mathcal{G}^n \times \mathbb{R}^n \mapsto \mathcal{G}$ that maps n element of group $\mathcal{G}$ weighted by $w \in \mathbb{R}^n$ to weighted mean values of element in $\mathcal{G}$. The mean operation is left-equivariant and invariance towards permutation [146], such that,

$$\mathcal{F}(g \circ \mathcal{X}, w) = g \circ \mathcal{F}(\mathcal{X}, w), \tag{5.4}$$

Due to transitivity property of equivariance [146] defined in **Definition 1**, the proposed network is rotation equivariance due to presence of equivariance network maps, i.e. PSN is equivariance as it is a local maximizer as defined in **Definition 2** and capsule network is equivariance due to its weighted mean operation in **Definition 3**.

5.2.1 Local Receptive Field Part Segmentation

Local receptive field is a common technique [170] in learning 2D images, where a 2D image is a set of pixel and each pixel is ordered in grid space. Regions of input space are employed to perceive local region of input for features encoding and learning. Intuitively, this allow sparse reception of input data as compared to fully connected networks, while able to extract significant features. On the contrary, a point cloud $S_1 = \{x_1, \dots, x_n\} \in \mathbb{R}^{N \times 3}$, with N number of points, is commonly presented in sparse and un-ordered grid-less space. Using intuition from local receptive field of 2D images, similar can be implemented by segmenting local regions of input point cloud. In order to segment parts from point cloud in a sparse and un-ordered environment, one can implement clustering technique, i.e. k-clustering to obtain patches of point cloud that represent local parts. However, such technique is heuristic and requires high number of iterations to converge on each cluster. In addition, each initialization of k-clustering is randomized, resulting the acquired local parts to be random in each network feed-forward.

To overcome this, PSN based on PNDE is devised based on Chapter 3 to consistently segment local parts from an input point cloud. Figure 5.2 shows the PSN denoted by $f_1 : \mathbb{R}^{N \times 3} \mapsto \mathbb{R}^{M \times k \times 3}$ and $P = f_1(S_1)$. A collection of local point cloud parts

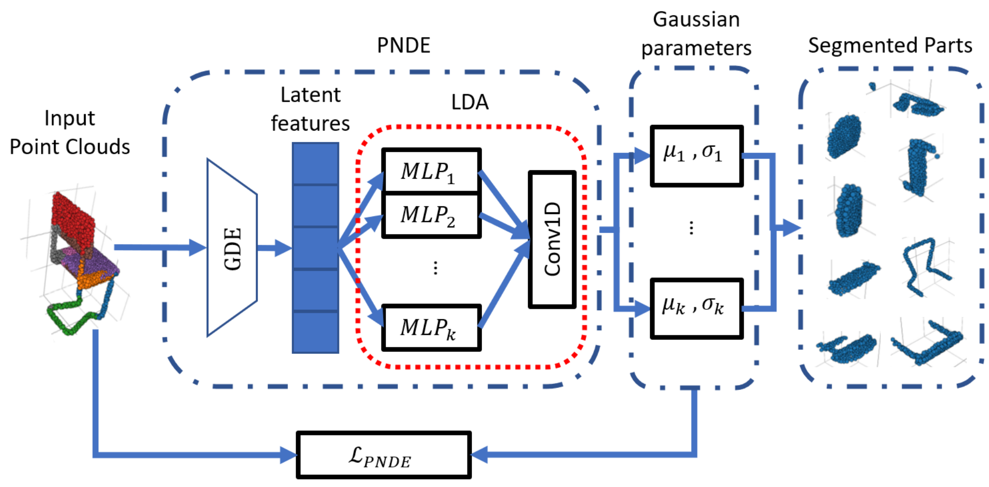


Figure 5.2: The PSN comprises PNDE with its components Global Density Estimator (GDE) and Local Density Autoencoder (LDA) to estimate its point cloud's mean and variance for local parts extraction using Gaussian maximum likelihood as an guiding objective function.

$P = \{p_1, \dots, p_M\} \in \mathbb{R}^{M \times k \times 3}$, where $k = \frac{N}{M}$ and M is number of density parameters, is sampled from input point cloud based on density parameters $Q = \{q_1, ...q_M\} \in \mathbb{R}^{M \times 6}$ generated from PNDE. By obtaining the density parameters of input point cloud, k highest likelihood points belonging to each density parameter are selected from the point cloud to form local point cloud part $p_m \in \mathbb{R}^{k \times 3}$. The procedure of PSN is described in **Procedure** 1.

Procedure 1 Part Sampler Network

Input: Input point cloud $S_1 = \{x_1, \dots, x_n\} \in \mathbb{R}^{N \times 3}$, Density parameters of input point cloud $Q = \{q_1, ...q_M\} \in \mathbb{R}^{M \times 6}$.

Output: Local parts of input point cloud $P = \{p_1, \dots, p_M\} \in \mathbb{R}^{M \times k \times 3}$, with $k = \frac{N}{M}$ number of points per local part point cloud and M number of density parameters.

```
1: for All density parameters $q_m$ do
2:     for All points of input point cloud $x_n$ do
3:         $l_{m,n} \leftarrow \mathcal{N}(x_n|q_m)$                ▷ Compute likelihood of each point
4:         $i_{m,k} \leftarrow \arg\max_{n=k}(l_{m,n})$            ▷ Search $k$ indices of highest likelihood
5:     end for
6:     $p_m \leftarrow select(S, i_{m,k})$        ▷ Select points from point cloud using the indices
7: end for
```

5.2.2 Part-to-Whole Learning using Capsule Network

Once the local point cloud parts are obtained, they are fed into a capsule network for part-to-whole relationship learning using a vector representation. In this implementation, the proposed capsule network comprises two main layers, i.e. parts capsules layer and class capsules layer. To obtain part capsules $U = \{u_1, \ldots, u_M\} \in \mathbb{R}^{M \times d}$, where d is the dimension size of a part capsule, a list of part encoder modules, $\{f_{enc,1}, \ldots, f_{enc,M}\}$ is adopted from PointNet [4] to encode individual part features, such that $f_{enc,m} : \mathbb{R}^{k \times 3} \mapsto \mathbb{R}^d$ and $u_m = f_{enc,m}(p_m)$. By normalizing the part capsule using a vector orientation-aware activation function [41] in Eq. (5.5), a part capsule holds the latent features of local point cloud part in the form of response vector.

$$squash(x) = \frac{||x||^2}{1 + ||x||^2} \frac{x}{||x||} \tag{5.5}$$

Subsequently, the class capsules $V = \{v_1, \ldots, v_C\} \in \mathbb{R}^{C \times d'}$, where d' is the dimension of a class capsule and C is the number of class, are responsible in containing features of each object class category. Different from standard perceptron operation, Dynamic Routing algorithm [41] is implemented to operate and propagate the part capsule to class capsule. The operation can be viewed as a weighted mean and soft clustering operation in voting each part capsule (lower-level features) into respective class capsule (higher-level features). As each class capsule represents a class category, the selection of class capsule to classify a point cloud is obtained by choosing the class capsule with most significant magnitude $v_{max} = V_{\underset{C}{\arg\max}(||V||)} \in \mathbb{R}^{d'}$. The procedure of the proposed part-based capsule network is described in **Procedure** 2. To further optimize the part capsules and latent feature of selected class capsule, they are concatenated to form a list of vectors $\{r_1, \ldots, r_M\} \in \mathbb{R}^{M \times (d+d')}$ for point cloud reconstruction. A patch-based decoder network adopted from Chapter 4 denoted by $f_{dec} = \{f_{dec,1}, \ldots, f_{dec,M}\}$, where $f_{dec,m} : \mathbb{R}^{d'+2} \mapsto \mathbb{R}^{k \times 3}$ is implemented to reconstruct output point cloud $S_2 = \{y_1, \ldots, y_N\} \in \mathbb{R}^{N \times 3}$.

5.2.3 Objective Functions

The optimization of the proposed network is conducted independently in the first stage and second stage. For instance, the PSN is optimized in prior to the capsule network, using Gaussian Maximum Likelihood as the objective function derived in Eq.

Procedure 2 Rotation Equivariance Part-based Capsule Network

```
Input: Local parts of input point cloud $P = \{p_1, \ldots, p_M\} \in \mathbb{R}^{M\times k\times 3}$, $r$ routing iterations.
Output: Class capsules $V = \{v_1, \ldots, v_C\} \in \mathbb{R}^{C\times d'}$.
 1: for All part capsule $m$ do
 2:     $u_m \leftarrow squash(f_{enc,m}(p_m))$                    ▷ Encode part capsule
 3: end for
 4: for All class capsule $c$ do
 5:     for All part capsule $m$ do
 6:         $\hat{u}_{c|m} \leftarrow W_{c,m}u_m$                   ▷ Transform part capsule
 7:     end for
 8: end for
 9: $b_{m,c} \leftarrow 0$                                       ▷ Initialize activation term
10: for $i$ routing iterations do
11:     for All part capsule $m$ do
12:         for All class capsule $c$ do
13:             $a_c \leftarrow softmax(b_m)$                   ▷ Normalize activation term
14:             $s_c \leftarrow \sum_c a_c\hat{u}_{c|m}$          ▷ Weighted average of new pose vector
15:             $v_c \leftarrow squash(s_c)$                    ▷ Normalize class capsule vector
16:             $b_{m,c} \leftarrow b_{m,c} + \hat{u}_{c|m} \cdot v_c$   ▷ Update activation term
17:         end for
18:     end for
19: end for
```

(5.6). This is to ensure the PSN is as generalized as possible and the part-to-whole learning is not dependent on the PNDE.

$$\mathcal{L}_{PNDE}(S_1, Q) = -\ln\left[\sum_{n=1}^{N}\prod_{m=1}^{M}\mathcal{N}(x_n|\mu_m, \Sigma_m)\right], \tag{5.6}$$

where,

$$\mathcal{N}(x_n|\mu_m, \Sigma_m) = \frac{1}{(2\pi)^{\frac{3}{2}}|\Sigma_m|^{\frac{1}{2}}}e^{-\frac{1}{2}(x_n-\mu_m)^T\Sigma_m^{-1}(x_n-\mu_m)}. \tag{5.7}$$

On the other hand, the optimization of capsule network in second stage uses a joint-loss objective function L_{cap}, where $L_{cap} = L_1 + L_2$. The first term of the joint-loss L_1 is the margin loss implemented in capsule network [41] as derived in Eq. (5.8),

$$L_1 = T_c\text{max}(0, m^+ - ||v_c||)^2 + \lambda(1 - T_c)\text{max}(0, ||v_c|| - m^-)^2 \tag{5.8}$$

where T_c is the targeted class and $T_c = 1$ if the object class exist. The margins $m^+ = 0.9$, $m^- = 0.1$, and the regularizing term is set $\lambda = 0.5$. The second term L_2 is Chamfer

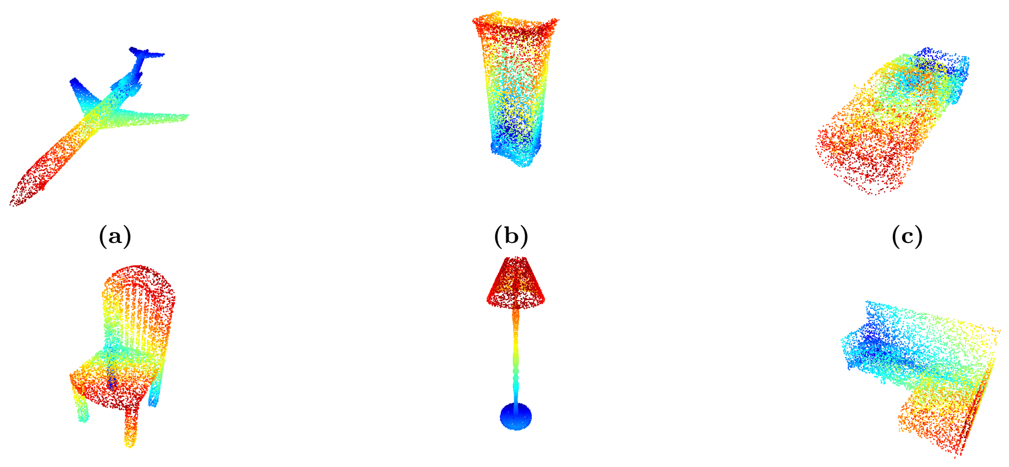


Figure 5.3: Visualization of dataset samples from Modelnet10, ModelNet40, and ShapeNetCore13: (a) Airplane, (b) Cabinet, (c) Car, (d) Chair, (e) Lamp, (f) Sofa.

distance (CD) [34, 35] that measures the reconstruction quality of reconstructed point cloud and it is derived in Eq. (5.9).

$$L_2(S_1, S_2) = \frac{1}{|S_1|} \sum_{x \in S_1} \min_{y \in S_2} ||x - y||_2^2 + \frac{1}{|S_2|} \sum_{y \in S_2} \min_{x \in S_1} ||x - y||_2^2, \tag{5.9}$$

5.3 Experimental Setup

The proposed network is evaluated using benchmark 3D point cloud datasets, i.e. ModelNet10, ModelNet40 [144] and ShapeNetCore13 [145] dataset. The ModelNet10 is a subset of ModelNet40 which contains 10 categories of labelled CAD object models with 3991 training and 908 testing data. On the other hand, ShapeNetCore13 is a subset of the full ShapeNetCore55 dataset, covering 13 common object categories of labelled CAD object models with 31,772 training and 7,956 testing data. The visualization of point cloud samples, i.e. airplane, cabinet, car, chair, lamp, and sofa are shown in Fig. 5.3. The proposed network is built using configuration shown in Table 5.1. Fixed hyper-parameters used in this implementation are class capsule dimension $d' = 64$, input point cloud size $N = 2048$, and routing iterations $i = 3$, class capsule size $C = 10, 40, 13$ for ModelNet10, ModelNet40, and ShapeNetCore13 respetively. Varied hyper-parameters are number of parts $M = 16, 32, 64, 128, 256, 512$ and primary capsule dimension $d = 256$ subjected in ablation studies. The proposed network is developed on

Table 5.1: Configuration of PSN, Capsule network and patch-based decoder.

Layers	Configuration	Output Dim.
Input	-	$N \times 3$
PSN	PNDE	$M \times 6$
	Likelihood Segmentation	$M \times k \times 3$
Part encoder modules	$M\times$ Conv1D $[3, 64]$ +BN+ReLU	$M \times k \times 64$
	$M\times$ Conv1D $[64, 128]$ +BN+ReLU	$M \times k \times 128$
	$M\times$ Conv1D $[128, d]$ +BN+ReLU	$M \times k \times d$
	Maxpool1D (k)	$M \times d$
	$M\times$ FC Layer $[d, d]$ +BN+ReLU	$M \times d$
	$M\times$ FC Layer $[d, d]$ +BN+ReLU	$M \times d$
Capsule Network	Primary Capsule	$M \times d$
	Dynamic Routing	-
	Class Capsule	$C \times d'$
Select class capsule	$V_{\underset{C}{\arg\max}(\|\|V\|\|)}$	$1 \times d'$
Patches	Expand($\frac{N}{M}$)	$\frac{N}{M} \times d'$
Patch-based decoder	$M\times$ Shared MLP $[(d'+2), 1024]$ +BN+ReLU	$M \times 1024 \times \frac{N}{M}$
	$M\times$ Shared MLP $[1024, 512]$ +BN+ReLU	$M \times 512 \times \frac{N}{M}$
	$M\times$ Shared MLP $[512, 256]$ +BN+ReLU	$M \times 256 \times \frac{N}{M}$
	$M\times$ Shared MLP $[256, 3]$ +BN+Tanh	$M \times 3 \times \frac{N}{M}$
-	Concatenation	$N \times 3$

Pytorch 1.9.0 using mixed precision network parameter. Batch size of 64 and learning rate 1×10^{-3} are used for network optimization. The experiments are conducted in respective sections: (Experiment #1) Ablation studies on the proposed network to reveal the performance under varied conditions, (Experiment #2) Evaluations of classification performance compared to existing works on non-rotated point clouds, (Experiment #3) Qualitative and quantitative evaluations of proposed network's performance on rotation perturbed point clouds.

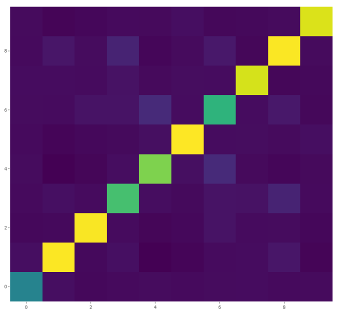
5.4 Experiment #1: Ablation Studies on Network Configuration

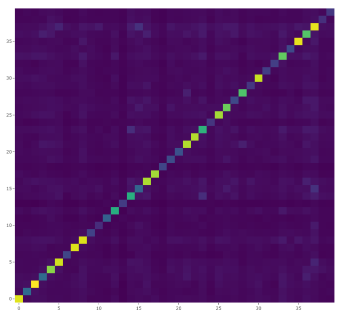
In this section, ablation studies are conducted on the proposed network using several benchmark datasets, i.e. ModelNet10, ModelNet40, and ShapeNetCore13. The ablation studies are aimed to reveal responses of proposed network toward several architectural variations. In addition, the studies can provide insights on optimized hyper-parameters of the proposed network. In Table 5.2, an ablation study on classification performance is performed on the proposed network by varying the number of local parts corresponding to the number part capsules and local point cloud receptive field. From the results, increasing the number of part capsules has minimal effect on the classification performance across all benchmark datasets. However, by increasing the size of local point cloud with factor of 2, an average of 0.4% improvement in classification performance is observed.

Essentially, this implies the size of local receptive field has more importance against the number of local parts presence for part whole learning. As the number of point per local part is defined by $N_p = \frac{N}{M} \times r$, where N is the number of point in a point cloud and r is the local receptive field size, it is inversely proportional to the number of parts. Thus, with fewer points presence in a local part, it translates to a local part that has reduced geometric information. The importance can be reaffirmed when slight improvement in the classification performance is observed as the local point cloud receptive field is increased by factor of 2.

Table 5.2: Network performance comparison on number of parts, M and receptive field size, r trained without reconstruction loss and evaluated using benchmark dataset ModelNet10, ModelNet40, and ShapeNetCore13 in classification rate (%).

#Parts	Model-Net10	Model-Net40	ShapeNet-Core13
$M = 16, r = 1$	92.67	87.47	94.19
$M = 32, r = 1$	92.67	87.72	94.25
$M = 64, r = 1$	92.67	88.25	94.00
$M = 16, r = 2$	93.11	88.00	94.33
$M = 32, r = 2$	93.11	87.85	94.58
$M = 64, r = 2$	93.44	88.50	94.26


(a)


(b)

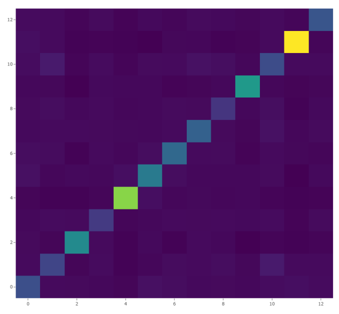

(c)

Figure 5.4: The cross-correlation matrix between class capsules on: (a) ModelNet10, (b) ModelNet40, (c) ShapeNetCore13.

Using configuration $M = 16$ and $r = 2$ of the proposed network, consecutive ablation study is conducted by incorporating reconstruction loss to optimize the class capsule. In Table 5.3, the results indicate that with additional guidance from reconstruction loss using Chamfer Distance, the classification performance of the proposed network is incrementally improved by an average of 0.5%. In Figure 5.4, correlation matrix of class capsules are visualized in form of confusion matrix to understand the relation of

Table 5.3: Network performance using configuration $M = 16, r = 2$ comparison with (w/) and without (w/o) reconstruction loss evaluated on benchmark dataset ModelNet10, ModelNet40, and ShapeNetCore13 in classification rate (%).

#Parts	Model-Net10	Model-Net40	ShapeNet-Core13
w/o CD	93.11	88.00	94.33
w/ CD	93.78	88.70	94.71

Table 5.4: Network parameters comparison using configuration $M = 16$, $r = 2$ and varied part capsule dimension evaluated on benchmark dataset ModelNet10, ModelNet40, and ShapeNetCore13 in classification rate (%).

Part Capsule Dim.	#param	Model-Net10	Model-Net40	ShapeNet-Core13
32	0.33M	93.56	88.62	94.71
64	0.66M	93.67	88.46	94.71
128	1.31M	93.56	88.25	94.84
256	2.62M	93.78	88.70	94.65
512	5.24M	93.78	88.70	94.64

class capsules and to inspect distribution of shape learning. From the visualization, it is observed classes with most confusion are ones with similar shape such as cabinet and night stand; flower pot and vase.

In Table 5.4, the proposed network is further studied on varied part capsule dimensions. From the results, the number of network parameter increases linearly to the part capsule dimension size with minimum of $0.33M$ trainable parameters when using $Dim = 32$ and maximum of $5.24M$ trainable parameters when using $Dim = 512$. However, with $16\times$ smaller network with $Dim = 32$, the degradation on classification performance is negligible. This shows the saturation in part capsules latent dimension, where with $Dim = 32$ the part capsules are sufficient to contain information required to represent a part as $Dim = 512$. From the ablation studies, design choices of the proposed network are made with the most optimized hyperparameter, i.e. $M = 16$, $r = 2$, $Dim = 32$ and incorporating reconstruction loss for network training.

5.5 Experiment #2: Point Clouds Classification Evaluation

In this section, the proposed network is compared to several pioneering works in point cloud classification. Table 5.5 shows the classification performance of proposed network and existing works on ModelNet10, ModelNet40, and ShapeNetCore13 to evaluate point clouds object learning capability. Despite using significantly less number of network parameters, the proposed network achieved on-par classification rate with 92.9% on ModelNet10, 88.9% on ModelNet40 and 94.1% on ShapeNetCore13. Notably,

Table 5.5: Network parameters comparison on the benchmark dataset ModelNet10, ModelNet40, and ShapeNetCore13 in classification rate (%).

Method	#param	Model-Net10	Model-Net40	ShapeNet-Core13
PointNet [4]	3.5M	92.7	89.4	94.5
PointNet++ [7]	1.5M	-	89.2	-
DGCNN [52]	1.8M	94.5	91.5	95.1
3DPointCapsule [104]	69.2M	92.4	89.3	93.9
Zhao et al. [48]	0.4M	-	85.3	-
Ours (32) + CD	0.33M	93.56	88.62	94.71
Ours (512)	5.24M	93.78	88.70	94.64

the proposed network is approximately 182 folds smaller than 3DPointCapsule while having similar classification performance.

To draw differences compared to existing works, the proposed network segments local parts of the input point clouds for parts reasoning through agreement via Dynamic Routing algorithm [41] to vote on an object. Essentially, the agreement to vote is achieved by searching most significant response of part capsules corresponding to an object capsule based on a dot-product operation. In contrary, most existing works rely on extending the dimension of latent features of the input point clouds to draw the most significant features. Although DGCNN outperformed all networks with an adequate network parameters, the added stacks of EdgeConv made the network computation extensive due to the search for neighbourhood points on each layer.

While the closely related work 3DPointCapsule implemented 1024 replicas of point cloud encoder to search for significant parts, it similarly implemented Dynamic Routing algorithm to vote on its lower level capsules corresponding to its higher level capsule. However, due to high numbers of replicated encoder, the network parameter is drastically increased. In contrast, the proposed network utilizes 16 segmented parts to encode into 16 part capsules, comparing to 1024 primary capsules in 3DPointCapsule. Hence, this justifies the efficacy of object learning through agreement on segmented parts compared to parts searching by network where it sees fit.

5.6 Experiment #3: Rotation Perturbed Point Clouds Classification and Reconstruction Evaluation

Rotation perturbation is critical due to its tendency to distort the input point clouds that can significantly affects performance in point clouds classification. To show the robustness of the proposed network on rotation perturbation, this section evaluates

Table 5.6: Network parameters comparison on the benchmark dataset ModelNet10, ModelNet40, and ShapeNetCore13 in classification rate (%).

Method	ModelNet10	ModelNet40	ShapeNet-Core13
Srivastava et al. [42]	96.0	–	–
Zhao et al. [48]	-	74.4	-
Ours (32) + CD	86.67	79.34	87.83

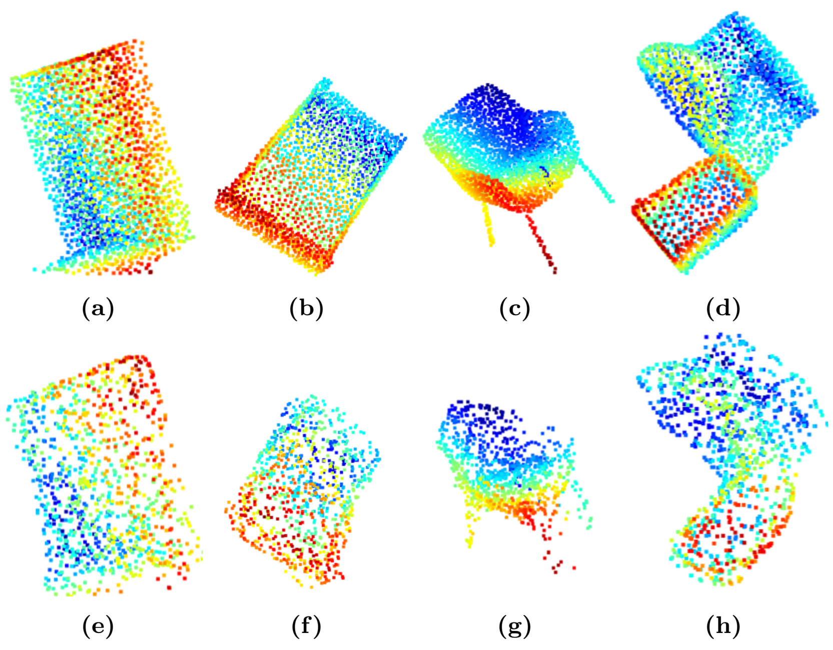


Figure 5.5: Qualitative results of point clouds reconstruction and rotation perturbed input point clouds: Ground truth (a) Bed, (b) Chair, (c) Dresser, (d) Toilet, Reconstructed point clouds (e) Bed, (f) Chair, (g) Dresser, (h) Toilet.

and discusses performance of point clouds classification on rotation perturbed input point clouds and comparison on existing works. In a closely related work [42], explicit pose estimation is implemented as part of viewpoint feature in a point cloud capsule network to achieve rotation equivariance. The work relies on drawing discrete novel viewpoints from an input point cloud and DR is utilized to vote on existence of object based on the novel viewpoints. Subsequently, the estimated pose is imposed on a reconstructed canonical point cloud to obtain rotation equivariance reconstruction. In more recent work [48], property of rotation equivariance is implicitly embedded in the network by utilizing collection of local reference frames and capsule network architecture. As local reference frame is a local orientation representation, hence it is invariant to global rotation. Therefore, rotation equivariant latent features can be generated by using DR that is a weighted averaging operation of the local reference frame.

In converse to existing works, the proposed method is rotation equivariance by joint network of rotation equivariance maps [146, 147], i.e. PSN and capsule network. Firstly, the PSN is a local maximizer function that is a transformation equivariant map, where no translation is involved in this work. Secondly, a capsule network architecture is an equivariant map by its weighted averaging operation. Furthermore, equivariance can be enforced when the local receptive field is not agnostic to the pose [146]. Here, the PSN generates a non-agnostic local receptive field by utilizing the point clouds density parameter that is equivariant to the input point clouds orientation. Hence, the extracted local part features are locally equivariance and globally invariance to transformation, i.e. rotation. To quantify the evaluation of rotation equivariance, Iterative Closest Point (ICP) [171] is used to measure the rotation discrepancies between ground truth point clouds and reconstructed point clouds. In the quantitative experiment, the reported average rotational deviation is $\pm 2\,\text{deg}$. The qualitative and quantitative results essentially evaluate the rotation equivariance property of the proposed network. In Fig. 5.5, the qualitative results show point clouds reconstruction from input point clouds imposed by random rotation of $[\frac{\pi}{2}, \pi]$ on the XYZ axes similar to setup in [42].

5.7 Chapter Summary

A capsule network with enforced part-to-whole relationship is proposed in this chapter to improve the effectiveness of point clouds learning. The network comprises two stages, i.e. (i) a PSN using Point cloud Neural Density Estimator (PNDE) as its backbone is proposed to segment local part of input point cloud; (ii) a capsule network is implemented to learn the reasoning of an object's existence using the agreement on parts voting through dynamic routing algorithm. Subsequently, a decoder inspired from Gaussian Point cloud Autoencoder (GPAE) is attached to the object capsules for generative point cloud reconstruction to further guide the optimization of object capsules. The proposed network showed an adequate point clouds object learning capability evaluated on classification performance compared to exiting works despite having significantly less network parameter. In addition, the proposed network requires significantly lower latent dimension in part capsule compared to existing works that use higher latent dimension, resulting in substantial reduced network parameter. Moreover, the proposed network is further improved with addition of reconstruction loss as the network joint loss function. From evaluation, the network achieved testing classification accuracy of 93.56% in ModelNet10, 88.70% in ModelNet40 and 94.71% in ShapeNetCore13. In evaluation where the input point clouds are imposed with rotation perturbation, the proposed network achieved test accuracy of 86.67% in ModelNet10, 79.34% in ModelNet40 and 87.83% in ShapeNet13. In qualitative results, the network is shown to fully reconstruct output that is equivariant to input point cloud with imposed rotation perturbation. Furthermore, the reconstructed output of the proposed network is quantitatively evaluated using ICP algorithm and achieved average $\pm 2\,\mathrm{deg}$ discrepancies in rotation. Nonetheless, this chapter showed the viability of proposed PNDE and decoder functioning as a generative learning model by incorporating part-to-whole learning architecture such as capsule network. In the following chapter, the proposed PNDE and decoder are further applied on real world application such as 3D human reconstruction to demonstrate the applicability of the networks.

Chapter 6

Skeletal Joints-based 3D Human Reconstruction from Partial Point Cloud

6.1 Introduction

The reconstruction of 3D human model from point cloud is of great interest in computer graphic and computer vision [16, 172] due to its wide applications, such as personalized human model in mixed reality applications [173], human kinematic measurements in sports coaching [174] and human modeling in video-based motion capture [17]. However, the acquisition of 3D human model from real-world is challenging mainly due to viewpoint occlusion [175], hence identifying body parts given a point cloud is often non-trivial. To overcome this, learning-based methods [176] are used to process sparse or incomplete point clouds, as they can leverage prior data to fill in the missing information in the input. Other deep learning models [177] predict body model parameters in feed-forward pass and use bottom-up two-dimensional (2D) features for self-supervision.

Reconstructing a high-fidelity human shape [19, 178] using single viewpoint is challenging because of non-rigid human deformations such as dynamic joints articulation and sparse partial input data. Emerging deep learning techniques made reconstruction of human shapes in an end-to-end fashion possible [179, 180]. Litany et al. [127] proposed the use of variational autoencoder incorporated with graph convolutional operations for the completion of partial shapes. It learns a latent space for complete realistic shape on full shapes with vertex-wise correspondence. Varol et al. [181] proposed a BodyNet to learn to reconstruct volumetric human shapes with a low resolution volumetric representation. On the other hand, Jiang et al. [16] proposed to incorporate skeleton awareness into the deep learning based regression of Skinned Multi-Person Linear model (SMPL) [123] parameters for 3D human shape reconstruction. The basic structure of this model uses PointNet++ to extract point features and then map point features to skeleton joint features and finally SMPL parameters for the reconstruction from point clouds. SMPL offers a nice compact representation for 3D human shape, and it has been integrated with deep neural networks for 3D human reconstruction from RGB images. Zhou et al. [17] performed the 3D human pose estimation from 2D depth images to 3D point clouds through convolutional neural networks (CNNs). The output of this network is the 3D coordinate of skeleton joints using pose regression network.

Numerous works in generative deep learning networks [34, 35, 102] are devised on single viewpoint for point clouds completion by leveraging the learning capability of deep learning neural networks. PCN [34], implemented a graph-based encoder coupled with FoldingNet decoder for point cloud completion learning. However, the implemented FoldingNet decoder in PCN involves multiple iterative processing that burdens the process to generate output point clouds. In contrary, MSN [35] proposed a coarse-to-fine grain point clouds sampling by introducing a novel minimum sampling strategy and implemented a patch-based decoder from [100]. The method out-performed PCN with a trade-off at the expense of higher network complexity. On the other hand, TopNet [102] proposed a hierarchical tree structure to generate structured point clouds by modeling point clouds topology. Although this allows a better generalization to novel shapes, the method is intractable to scalability and has limited shapes learning capacity due to the dependency of operating nodes in the tree structure.

Inspired from recent research trend in point cloud reconstruction using autoencoder [35, 37] and regressive 3D human model-based reconstruction [16, 124], two 3D human point cloud reconstruction models are proposed in this chapter and are sectioned in Section 6.2 and Section 6.3. The aim of the proposed models is to obtain complete point cloud of human by taking input of sparse partial point cloud acquired from depth sensor. In Section 6.2, a Generative skeletal joint-based autoencoder for human point cloud reconstruction model is proposed. The model is a generative autoencoder model that reconstructs point cloud of human from input partial point cloud acquired from a single viewpoint depth sensor, i.e. stereo vision camera. As a generative autoencoder model, the output product is a non-synthetic complete point cloud reconstructed from the input partial point cloud. In the following, a synthetic skeletal joints-based regressive human reconstruction model is proposed in Section 6.3 with an incremental network revision that employs more efficient encoder and a synthetic human decoder. Hence, output product of the synthetic model is a synthetic 3D human constructed from meshes with concise representation of human shape. Subsequently, complete point cloud of human is sampled directly from the surface of the synthetic human model. Both qualitative and quantitative experiments are conducted using real-world non-synthetic human dataset to demonstrate the feasibility of the proposed models.

6.2 Generative Skeletal Joint-based Autoencoder for 3D Human Point Cloud Reconstruction

Point cloud acquired from depth sensors often possess attributes of un-ordered and non-uniform data distribution in the 3D space. In addition, single viewpoint acquisition may incur occlusion of human parts resulting in partial point clouds and absence of crucial information from acquisition subject that can greatly degrade the performance of human model reconstruction and skeletal joints estimation. Recent autoencoder architectures [35, 37] have established significant qualitative improvements and feasibility of the architecture in the works of point cloud reconstruction. Inspired by the architecture, an autoencoder for 3D human point cloud reconstruction from input sparse partial

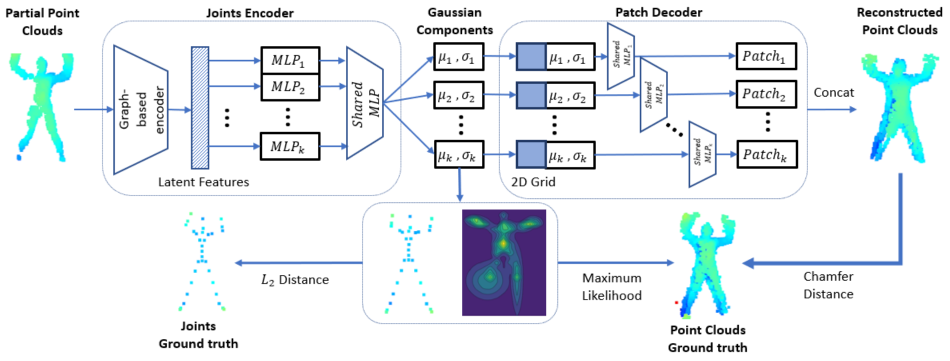


Figure 6.1: The proposed network comprises two-stage operations: (1) Joints encoder, (2) Patch-based decoder. Joint loss of L_2 distance, maximum likelihood and Chamfer distance are implemented to optimize the network.

point clouds acquired from a single viewpoint depth sensor is proposed as shown in Fig. 6.1.

The proposed autoencoder comprises two stages: (i) Skeletal joints encoder and (ii) Patch decoder. The skeletal joints encoder learns to encodes latent representation of input partial point clouds using graph-based features [52]. Using learnt latent representation of partial point clouds, a set of localized skeletal joints and its variances are inferred by independent multi-layer perceptrons (MLPs). Intuitively, the skeletal joints and its variances can be viewed as local human part keypoints and subsequently utilized for local patch sampling using the patch decoder adopted from AtlasNet [100]. By concatenating the sampled local patches, a full 3D human point cloud is reconstructed.

6.2.1 Stage 1: Skeletal Joints Encoder

Let a set of full point cloud of a human defined as $S = \{p_i \in \mathbb{R}^3\}_{i=1}^{N}$, where N is the number of points. A set of input partial point cloud $S_{par} = \{p_i \in \mathbb{R}^3\}_{i=1}^{N_{par}}$ is defined as the subset of the complete point cloud, such that $S_{par} \subseteq S$ and $N_{par} < N$. Given there exists K ground truth skeletal joints $\{\mu_{gt,k}\}_{k=1}^{K}$ from the human, then K estimated skeletal joint components $\{\mu_k, \Sigma_k\}_{k=1}^{K}$ can be inferred from the input partial point cloud. Due to the skeletal joints components are modeled after mean and variance, an encoder network such as network architecture laid out in Chapter 3 can be deployed as a Skeletal Joints Encoder (SJE).

SJE is a network devised to estimate skeletal joints and joint variances from input partial point cloud of a human. The network comprises a graph-based encoder adopted from [52], a set of K independent MLPs and a shared MLP, where K is the number of skeletal joint. A graph-based encoder is chosen to encode the latent features of input point cloud due to its capability in aggregating semantic point clouds features relative to its neighbourhood points. Moreover, an encoded latent feature gives a reduced dimensionality of input data, which allow faster and efficient latent inference using MLPs. Next, K independent MLPs and a shared MLP are implemented to infer independent skeletal joints and joint variances. Mean square error (MSE) of skeletal joints in Eq. (6.1) is used to compute the loss between estimated skeletal joints and ground truth skeletal joints to guide the parameter optimization of the encoder. Additionally, a Gaussian maximum likelihood function in Eq. (6.2) is used to further improve the estimation by maximizing the likelihood of estimated skeletal joints and joint variances respective to complete point cloud.

$$d_{joints}(\mu, \mu_{gt}) = \frac{1}{K}\sum_{k=1}^{K} ||\mu_k - \mu_{gt,k}||_2^2, \tag{6.1}$$

$$p(S|\mu, \sigma) = \sum_{i=1}^{N}\prod_{k=1}^{K} \mathcal{N}(p_i|\mu_k, \sigma_k), \tag{6.2}$$

where,

$$\mathcal{N}(p_i|\mu_k, \sigma_k) = \frac{1}{(2\pi)^{\frac{3}{2}}|\sigma_k|^{\frac{1}{2}}} e^{-\frac{1}{2}(p_i-\mu_k)^T\sigma_k^{-1}(p_i-\mu_k)}. \tag{6.3}$$

6.2.2 Stage 2: Patch Decoder

To reconstruct a complete human point cloud subsequent to the inference from SJE, skeletal components are used as input parameter for a patch decoder for generative reconstruction. This is reasoned by viewing the composition of skeletal components as the local human part representation, where the mean components represent localized skeletal joints and the variance components represent distribution of point cloud of local human parts. Intuitively, a generative sampling technique such as one proposed in Chapter 4 can be applied to reconstruct complete human point cloud by supplying skeletal components to a patch decoder, such that,

$$\hat{p}_i \sim \mathcal{N}(\mu_k, \Sigma_k), \tag{6.4}$$

where the reconstructed set of point clouds is defined as $\hat{S} = \{\hat{p}_i \in \mathbb{R}^3\}_{i=1}^{N}$.

Following network architecture laid out in Chapter 4, a 3D point cloud reconstruction approach is implemented to sample multiple local patches of point cloud using patch decoder comprises independent shared MLPs. Subsequently, the local patches are assembled to obtain a complete 3D point cloud by concatenation. In the point cloud reconstruction approach, each feature component, i.e. Gaussian, is used as a basis function to guide an independent shared MLP for local 2D grid deformation. Intuitively, each skeletal components is viewed as similar input component, hence local 2D grid deformation can be implemented to reconstruct a complete human point cloud. To optimize the patch decoder, Chamfer distance (CD) [109] denoted as $d_{CD}(\cdot)$ is implemented as the reconstruction loss and is defined as follows,

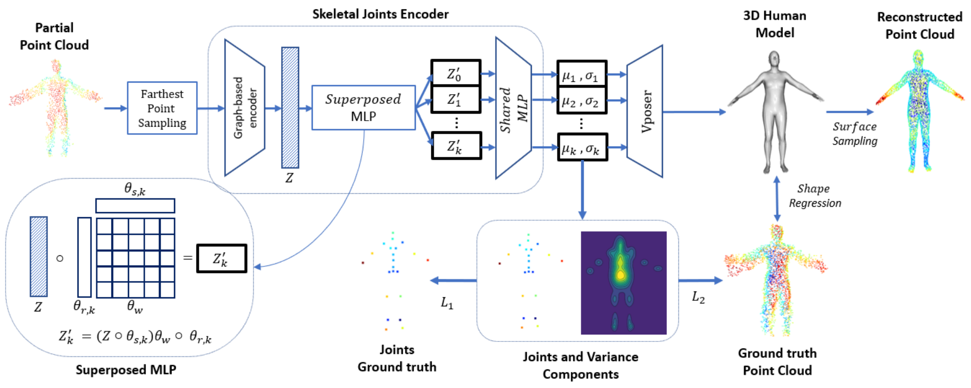


Figure 6.2: The proposed skeletal joints-based regressive human reconstruction network is trained using two-mode strategy, i.e. synthetic training and fine-tuning on real-world dataset.

$$d_{CD}(\hat{S}, S) = \frac{1}{|\hat{S}|} \sum_{x \in \hat{S}} \min_{y \in S} ||x - y||_2^2 + \frac{1}{|S|} \sum_{y \in S} \min_{x \in \hat{S}} ||x - y||_2^2, \tag{6.5}$$

where the similarities are the Euclidean distance of two nearest point and the computation can be efficiently implemented in parallel processing while retaining permutation invariant. In order to draw a difference of the loss function compared to Chapter 4, mean squared error (MSE) between ground truth and estimated skeletal joints $d_{joints}(\mu, \mu_{gt})$ is incorporated to further improve the guidance of the estimation. Therefore, the overall loss function for network end-to-end optimization is defined as follows,

$$d_{joints}(\mu, \mu_{gt}) = \frac{1}{K} \sum_{k=1}^{K} ||\mu_k - \mu_{gt,k}||_2^2, \tag{6.6}$$

$$\mathcal{L} = d_{joints}(\mu, \mu_{gt}) - \ln(p(S|\mu, \sigma)) + d_{CD}(\hat{S}, S). \tag{6.7}$$

6.3 Skeletal Joints-based Regressive Synthetic 3D Human Reconstruction

A skeletal joints-based regressive 3D human point cloud reconstruction is proposed to reconstruct 3D human model using two-stage operations as illustrated in 6.2. It includes a skeletal joints encoder to determine localized skeletal joints and variances such as one implemented in aforementioned generative model in Section 6.2, and a regressive synthetic human model generator to obtain synthetic 3D human model. By sampling on the surface of the synthetic 3D human model, a complete human point cloud can be obtained. Due to the nature of non-uniform surface and irregular density data distribution of acquired partial point cloud [50], farthest point sampling (FPS) algorithm [182] is implemented in the pre-processing step to uniformly distribute input partial point clouds in a grid-less space. Subsequently, a skeletal joint encoder from Section 6.2 and parameter compression strategy from Chapter 3 are employed to infer K skeletal joints and joint variances from the uniformly resampled input partial point cloud.

The proposed skeletal joints encoder includes a graph-based encoder [52] and a superposed MLP to infer the skeletal joints components. Compared to stack MLPs implemented in Section 6.2, MLPs in the proposed network are vectorized into matrix operation as demonstrated in Chapter 3. Consequently, the vectorization of MLP into matrices eliminate the iterations operation in stack MLPs resulting in largely reduced forward propagation time. Following by the matrix vectorization, a parameters superposition compression strategy [143] is adopted to decompose the vectorized matrices, i.e. $\theta_w \in \mathbb{R}^{d\times d'}, \theta_r \in \mathbb{R}^{K\times d'}, \theta_s \in \mathbb{R}^{K\times d}$, effectively reduces the memory footprint of overall network. By obtaining the estimated skeletal joints and variances, the estimated joints are fed into Vposer [183], a fully differentiable regressive synthetic human model generator that directly reconstruct a synthetic human model from human joints through inverse kinematics. Finally, surface sampling using Barycentric coordinate interpolation [184] is implemented to obtain reconstructed 3D human point cloud $S_3 = \{p_i \in \mathbb{R}^3\}_{i=1}^{N}$.

6.3.1 Two-mode Synthetic and Non-Synthetic Training

In order to achieve skeletal joints estimation and 3D human reconstruction from partial point cloud, a large dataset of partial point clouds that covers vast majority of possible human pose is required for network training. Moreover, it is resource expensive to acquire complete point clouds and skeletal joints of the corresponding partial point clouds. Existing works [16] on network training directly used real-world dataset as their training and testing data. However, method in [16] is only applicable for complete point clouds inference. While output of the generative reconstruction is non-synthetic, it may carry forward noises that are present in training data and less expressive in modeling a 3D human such as body shape.

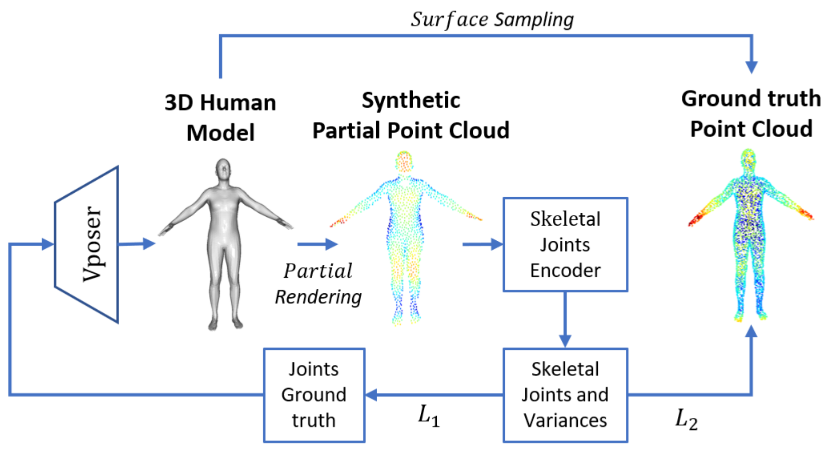


Figure 6.3: The training of the proposed skeletal joint encoder on synthetic partial point cloud rendered from Vposer 3D human model regressed on ground truth joints.

To overcome the aforementioned challenges, a two-mode training phase is proposed in the model training as illustrated in Fig. 6.3, where the first training phase learns general human shape and pose from synthetic partial point cloud synthesised from Vposer [183] regressed from ground truth joints. The synthetic partial point cloud is partially rendered using rendering technique from [34] on single-viewpoint. The second training phase fine-tunes the proposed network by using real-world training data as input and guided by ground truth joints. A set of scaling and shape parameters are regressed in the fine-tuning process to capture the human size and shape of real-world data. This is to further adapt the proposed method toward non-synthetic nature of real-world data captured from a depth sensor.

The network training uses weakly supervised joint loss $L = L_1 + L_2$ with mean squared error (MSE) shown in Eq. (6.8) and Gaussian maximum likelihood shown in Eq. (4.2) as follows,

$$L_1(\mu, \mu_{gt}) = \frac{1}{K}\sum_{k=1}^{K} ||\mu_k - \mu_{gt,k}||_2^2, \tag{6.8}$$

$$L_2(S_2|\mu, \sigma) = \sum_{i=1}^{N}\prod_{k=1}^{K} \mathcal{N}(p_i|\mu_k, \sigma_k), \tag{6.9}$$

where,

$$\mathcal{N}(p_i|\mu_k, \sigma_k) = \frac{1}{(2\pi)^{\frac{3}{2}}|\sigma_k|^{\frac{1}{2}}} e^{-\frac{1}{2}(p_i-\mu_k)^T\sigma_k^{-1}(p_i-\mu_k)}. \tag{6.10}$$

The MSE of skeletal joints is computed between the estimated skeletal joints with the ground truth to optimize encoder's parameter learning. Gaussian maximum likelihood function is implemented to improve the estimation by maximizing the likelihood of estimated skeletal joints and its variances respective to complete point clouds.

6.4 Experiment #1: Generative 3D Human Reconstruction

The experimental evaluation of the proposed generative model is based on Berkeley Multimodal Human Action Database (MHAD) [185] dataset with similar pre-processing setting as stated in [16]. MHAD is a non-synthetic 3D human model dataset that contains 11 actions performed by 12 human subjects. All subjects performed 5 repetitions of each action, yielding 660 action sequences which correspond to averagely 200 frames per action sequence. The dataset is split into 4 repetitions for training set and 1 repetition for testing set for each subject. In this experiment, Seq # 1 (jumping) and Seq #2 (jumping jack) are used to perform evaluation to demonstrate the working principle of the proposed network. There are 9280 training samples and 2368 testing samples for network optimization. Each sample is normalized into $[-1,1]^3$ unit sphere using the centroid of bounding box defined by the minimum and maximum joint positions.

Table 6.1: Configuration of generative 3D human reconstruction network.

Layers	Configuration	Output Dim.
Input	-	$N_{par} \times 3$
Skeletal joints encoder	EdgeConv ($k = 10$) $[3, 64, 128, 128]$	$N_{par} \times 128$
	Conv1D $[128, 256]$ +BN+ReLU	$N_{par} \times 256$
	Conv1D $[256, 1024]$ +BN+ReLU	$N_{par} \times d$
	Maxpool1D (N_{par})	$1 \times d$
	(Stacked MLP) $K\times$ FC Layer $[d \times d']$ +BN+ReLU	$K \times 512$
	Conv1D $[512, 256]$ +BN+ReLU	$K \times 256$
	Conv1D $[256, 6]$ +BN	$K \times 6$
Gaussian Components	ReLU ($K \times 3$)	$K \times 6$
	Softplus ($K \times 3$)	
Patches	Expand($\frac{N}{K}$)	$K \times \frac{N}{K} \times 6$
Patch decoder	$K\times$ Shared MLP $[(6 + 2), 1024]$ +BN+ReLU	$K \times \frac{N}{K} \times 1024$
	$K\times$ Shared MLP $[1024, 512]$ +BN+ReLU	$K \times \frac{N}{K} \times 512$
	$K\times$ Shared MLP $[512, 256]$ +BN+ReLU	$K \times \frac{N}{K} \times 256$
	$K\times$ Shared MLP $[256, 3]$ +BN+Tanh	$K \times \frac{N}{K} \times 3$
-	Concatenation	$N \times 3$

In this experiment, the number of skeletal joints is set $K = 35$ as defined by MHAD dataset, the number partial point clouds is set $N_{par} = 2048$, number of ground truth point clouds and reconstructed point clouds is set $N = 6890$ for fair comparison against existing work. The architectural details are shown in Table 6.1 with latent features dimension $d = 1024$ and $d' = 512$. For the evaluation metrics, the average point-to-vertex distance d_{p2v} in Eq. (6.11) and average vertex-to-point distance d_{v2p} in Eq. (6.12) is used to evaluate the precision and recall performance of the reconstructed human point clouds and ground truth point clouds.

$$d_{p2v}(\hat{S}, S) = \frac{1}{|\hat{S}|} \sum_{x \in \hat{S}} \min_{y \in S} ||x - y||_2^2 \tag{6.11}$$

$$d_{v2p}(\hat{S}, S) = \frac{1}{|S|} \sum_{y \in S} \min_{x \in \hat{S}} ||x - y||_2^2, \tag{6.12}$$

Experiments are trained 200 epochs with scheduling decay rate of 0.5 per 20 epoch and starting learning rate is set 1×10^{-3}. Parameters are initialized using Xavier normal

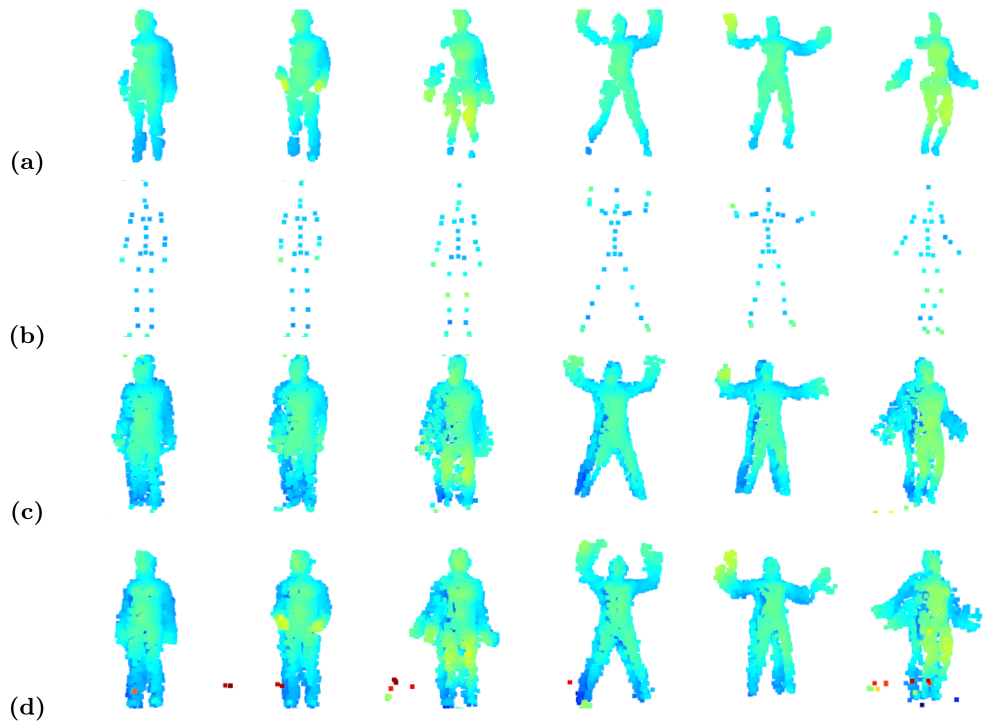


Figure 6.4: These human model are captured in (a) the partial input point clouds , (b) the estimated skeletal joints using partial input point clouds, (c) the reconstructed 3D human point clouds, and (d) the ground truth 3D human point clouds .

and batch size 64 is used. ADAM optimizer is implemented as optimization method for both parts of training. The proposed networks are built and executed on Pytorch 1.8.0. The specifications of test bench for the experiments are Intel-i7-4790K with 32GB RAM and Quadro P6000 GPU with 24GB VRAM.

The demonstration of skeletal joints estimation and 3D human point clouds reconstruction is shown in Fig. 6.4 on a sequence of human motion. Fig. 6.4(a) illustrates the input partial point clouds that is acquired from single viewpoint depth sensing camera. As these partial point clouds are fed into the network, the joints encoder retrieves K estimated skeletal joints as shown in Fig. 6.4(b). Subsequently, patch decoder decodes the estimated joints to reconstruct full point clouds as displayed in Fig. 6.4(c). In Fig. 6.4(d), the ground truth point clouds are captured in full form from two depth sensing cameras placed opposing each other and this ground truth acts as the benchmark model to evaluate the reconstructed model. The implication of the illustration indicates that the proposed network is able to reconstruct a full 3D human point clouds by using single viewpoint acquisition that is on-par with the results from using multiple viewpoint acquisition.

Table 6.2 shows the average Chamfer distance and average joint distance (mm) of reconstructed 3D human point clouds over the 12 subjects. In the case of without maximum likelihood during joints encoder optimization, the encoder achieved an average joint distance deviation of $54.77mm$ and the decoder achieved an average Chamfer distance of 0.94×10^{-3}. By incorporating a maximum likelihood of estimated joints and its variance on ground truth point clouds, the proposed network achieved significantly better results with an average joint distance deviation of $32.17mm$ and an average Chamfer distance of 0.62×10^{-3}. This suggests that by using addition joint variances as shown in the variance map in Fig. 6.2, the proposed model is able to capture more semantic local part features. As the results, the reconstruction fidelity and estimation of skeletal joints are greatly enhanced. Moreover, the proposed model also showed robustness and generalization towards sparse input partial point clouds as shown in Fig. 6.5. The figure

Table 6.2: The average estimated joints deviation against ground truth joints in mm and average reconstruction loss d_{CD} with and without maximum likelihood (ML).

Methods	Joints Deviation (mm)	d_{CD} ($\times 10^{-3}$)
w/o ML	54.77	0.94
w/ ML	**32.17**	**0.62**

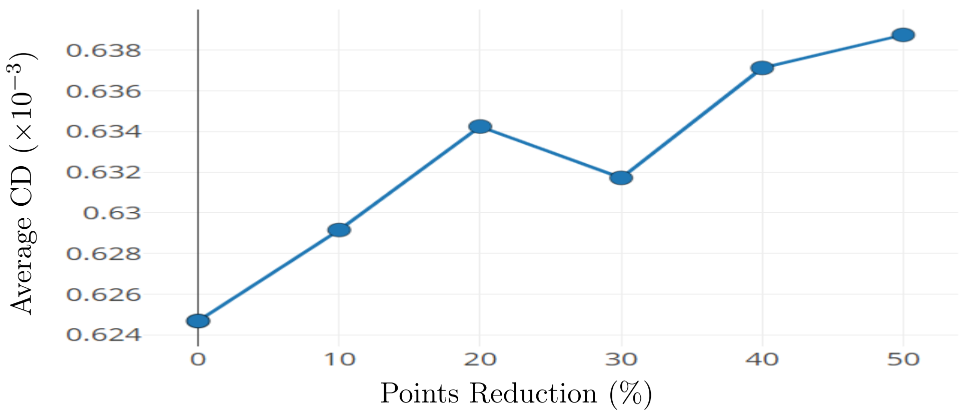


Figure 6.5: Average CD respective to points reduction (%) in the input partial point clouds on two motion sequences.

shows an approximate of 2% degradation in reconstruction performance with up to 50% points reduction on the input partial point clouds.

Comparing to the existing point clouds reconstruction architecture, single latent features that govern a full point clouds basis function is given to decoder to generate output point clouds. A point cloud completion method proposed by Liu et al. [35] out-performed other methods [34, 100] due to the coarse-to-fine completion architecture that used larger number of trainable parameters. However, a set of point clouds is governed by multiple basis function which defines the manifold of a full point clouds [104, 154]. Intuitively, the proposed network uses K number of joints and its variances as the governing basis function for each local patch of full 3D human point clouds. This gave an advantage to each patch decoder to avoid ambiguity and redundancy when generating point clouds.

A notable closely related work involved in 3D human point clouds [16] implemented a three stage mechanism in learning input 3D human point clouds. The approach learns to generate a SMPL parameters for synthetic 3D human reconstruction using 3D human point clouds as input. The first stage of the approach implements a point-wise learning network from [7]. The second stage implements a graph feature extraction on the points features from first stage to aggregate relationship between neighbourhood points and decode the skeletal joints. Lastly, an attention network module is implemented for decoded joints ordering. Subsequently, a set of SMPL parameters are regressed using the learnt graph features for synthetic SMPL 3D human reconstruction. In converse, the proposed network directly implements a graph-based point learning from [52]. Next,

Table 6.3: Mean and max distance (mm) measurement on two motion sequences in Berkeley MHAD dataset [185]. Note that in each cell, the first and second numbers denote the distances d_{p2v}/d_{v2p} respectively.

Methods	Seq #1		Seq #2		#param ($\times 10^6$)
	mean	max	mean	max	
Jiang et al. [16]	21.4/23.5	28.6/34.7	16.9/18.2	21.5/21.2	5.03
Ours	20.37/24.49	49.92/34.03	24.16/21.23	28.41/32.94	1.51

independent MLPs are used for latent inference which directly preserve the locality of joints output. Subsequently, the estimated joints and its variances generated by the proposed network is treated as the local part features of a 3D human model. Hence, a full 3D human point clouds can be directly sampled on the local part features.

Two sequences of motion, i.e. jumping and jumping jack are evaluated between the proposed model and [16] as tabulated in Table 6.3. Despite the measurement of mean and max distance value, the proposed method showed an adequate performance compared to the existing work [16]. However, as the existing work relies on external model, i.e. SMPL to produce a 3D human mesh prior to producing a 3D human point clouds, the existing work's network complexity and parameters are significantly higher. Furthermore, the existing work is immune towards outliers due to the nature of mesh-to-point clouds generation. In converse, the proposed model is a standalone network model that produces a full 3D human point clouds from raw input partial point clouds in an end-to-end fashion. As the consequence, the outliers as shown in Fig. 6.4d can contribute in the mean and max value deviation in the evaluation. Therefore, the proposed model needs to rely on pre-processing and post-processing to avoid the outliers.

6.5 Experiment #2: Regressive Synthetic 3D Human Reconstruction

In this experiment, the evaluation of the proposed synthetic model is based on Berkeley Multimodal Human Action Database (MHAD) [185] dataset with identical setup in Experiment #1 in section 6.4. The number of skeletal joints is set $K = 22$ as defined by Vposer. The number of points of point clouds are set $N_{par} = 1024$, $N = 6890$. The architectural details are shown in Table 6.4 and latent feature dimension is set $d = 1024$ and $d' = 512$. For the evaluation metrics, the average point-to-vertex distance

Table 6.4: Configuration of regressive 3D human reconstruction network.

Layers	Configuration	Output Dim.
Input	-	$N_{par} \times 3$
Skeletal joints encoder	EdgeConv ($k = 10$) $[3, 64, 128, 128]$	$N_{par} \times 128$
	Conv1D $[128, 256]$ +BN+ReLU	$N_{par} \times 256$
	Conv1D $[256, 1024]$ +BN+ReLU	$N_{par} \times 1024$
	Maxpool1D (N_{par})	1×1024
	(Superposed MLP) $(1024 \times 512) + (1024 \times K) + (512 \times K)$ +BN+ReLU	$K \times 512$
	Conv1D $[512, 256]$ +BN+ReLU	$K \times 256$
	Conv1D $[256, 6]$ +BN	$K \times 6$
Gaussian Components	ReLU ($K \times 3$)	$K \times 6$
	Softplus ($K \times 3$)	
Vposer	-	3D Human Model
Surface Sampling	-	$N \times 3$

d_{p2v} in Eq. (6.13) and average vertex-to-point distance d_{v2p} in Eq. (6.14) are used to evaluate the precision and recall performance of the reconstructed human vertices and ground truth point cloud. Lastly, the average Chamfer distance $d_{CD} = d_{p2v} + d_{v2p}$ is used to evaluate the reconstruction quality between ground truth and reconstructed point clouds.

$$d_{p2v}(S_2, S_3) = \frac{1}{|S_2|} \sum_{x \in S_2} \min_{y \in S_3} ||x - y||_2^2 \tag{6.13}$$

$$d_{v2p}(S_2, S_3) = \frac{1}{|S_3|} \sum_{y \in S_3} \min_{x \in S_2} ||x - y||_2^2, \tag{6.14}$$

The training of the proposed network is set for 100 epochs with scheduling decay rate of 0.5 per 20 epoch and starting learning rate is set 1×10^{-4}. Parameters are initialized using Xavier normal and batch size 64 is used. ADAM optimizer is implemented as optimization method for both parts of training. Meanwhile, the fine-tuning is set for 5 epoch with scheduling decay rate of 0.5 per 20 epoch and starting learning rate is set 1×10^{-5}. All networks are built and executed on Pytorch 1.8.0 with batch size of 128. The specifications of test bench for the experiments are Intel-i7-4790K with 32GB RAM and Quadro P6000 GPU with 24GB VRAM.

The qualitative results of the proposed method in complete 3D human point cloud reconstruction from real-world non-synthetic partial point clouds is demonstrated in Fig. 6.6. Fig. 6.6(a) illustrates a sequence of partial point clouds acquired from single-viewpoint depth sensor. Subsequently, by feeding the estimated skeletal joints illustrated in Fig. 6.6(b) into Vposer, 3D human models are reconstructed illustrated in Fig. 6.6(c). Through surface sampling on the reconstructed 3D human model, the

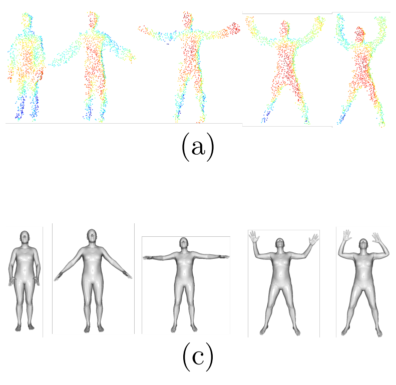
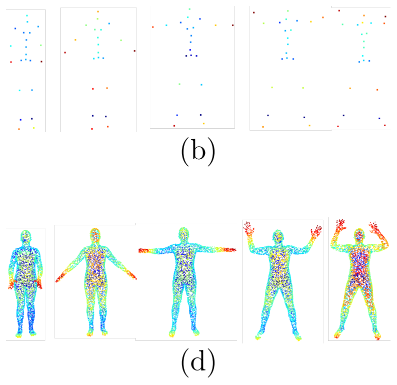



Figure 6.6: Qualitative results of the proposed network (a) the input partial point cloud from real-time, (b) the estimated skeletal joints using input partial point cloud, (c) the 3D human mesh, and (d) full reconstructed human point cloud.

complete human point cloud is obtained as illustrated in Fig. 6.6(d). Due to the surface sampling process, the surface quality of complete human point cloud is significantly superior compared to generative reconstruction. Moreover, noises such as outliers that can occur during reconstruction from a decoder is avoided.

Table 6.5 shows the evaluation of average joints deviation and average reconstruction loss comparing the proposed method in generative reconstruction. From the evaluation, the proposed method with fine tuning outperformed the existing work with $24.83mm$ in joints deviation compared to $45.29mm$. Moreover, a significant reconstruction quality improvement is shown with lower Chamfer distance of 0.84×10^{-3} compared to 1.0×10^{-3}. Notably, the improvement in average joints deviation is contributed by learning on synthetic data in prior which contain more concise information of partial point clouds. Furthermore, the improvement of reconstruction quality is due to uniform surface sampling of complete point cloud. As oppose to generative reconstruction , the technique can capture the artifacts such as non-uniform surface and outliers in training data and reproduce the artifacts in the output. In the case of the proposed method without fine tuning, the joint deviation and reconstruction loss are significantly higher. This is because the network is not adapted to non-synthetic data modality when solely trained on synthetic data.

Table 6.6 shows the quantitative results of mean and max distance measurements on Action #1 and Action #2 in Berkeley MHAD dataset [185] comparing to existing works. The evaluation is based on the point-to-vertex d_{p2v} and vertex-to-point d_{v2p} metric comparing ground truth and reconstructed point clouds. Essentially, the evaluation measures the quality of human shape reconstruction. The proposed method with fine-tuning outperformed the generative reconstruction in overall d_{p2v} and d_{v2p}. This indicates superior shape reconstruction as compared to decoder-based reconstruction. Without fine-tuning, the proposed method has significantly larger shape deviation compared to ground truth point clouds as indicated by large max value of d_{p2v} and d_{v2p}. The large shape deviation is mainly because the model does not estimate the human shape. On the other hand, other existing work such as [124] first estimates the skeletal joints using third party model to regress a 3D human model, while [16] uses complete point clouds to regress a 3D human model. Although the existing methods [16, 124] do not infer partially occluded data, they are able to reconstruct more precise human shape.

6.6 Chapter Summary

Two deep learning models on point cloud-based human reconstruction are proposed: (i) generative non-synthetic model, and (ii) regressive synthetic model to obtain complete representation of 3D human by taking input sparse partial point cloud acquired from single viewpoint depth camera. Both models share similar initial stage in the inference process, where skeletal joints are first inferred from input partial point cloud. The subsequent process of the models differs in generation of output such that generative model outputs complete point cloud of human using a patch-decoder, and the synthetic model outputs synthetic 3D human model using a synthetic human model

Table 6.5: The average estimated joints deviation against ground truth joints in millimeter (mm) and average reconstruction loss d_{CD} with (w/) and without (w/o) fine tuning (FT).

Methods	Joints Deviation (mm)	d_{CD} ($\times 10^{-3}$)
Generative reconstruction	45.29	1.0
Ours (w/o FT)	71.27	1.29
Ours (w/ FT)	**24.83**	**0.84**

Table 6.6: Quantitative results of mean and max distance in millimeter (mm) measurements on two action sequences in Berkeley MHAD dataset [185]. Note that in each cell, the first and second numbers denote the distances d_{p2v}/d_{v2p} respectively.

Methods	Action #1		Action #2	
	mean	max	mean	max
SMPLify [124]	31.1/41.1	43.4/58.8	31.3/39.7	48.6/58.4
Jiang et al. [16]	21.4/23.5	28.6/34.7	16.9/18.2	21.5/21.2
Generative model	40.99/46.75	72.91/78.54	45.92/46.09	97.21/63.80
Ours (w/o FT)	37.33/43.70	60.84/85.74	33.81/43.76	41.64/65.78
Ours (w/ FT)	34.32/30.53	47.07/44.13	34.78/35.29	48.21/55.46

regressor. The generative model is a generative skeletal joint-based autoencoder network that reconstructs complete point cloud of human by training using non-synthetic data, i.e. point clouds acquired from real-world. With addition of Gaussian maximum likelihood, the network is able to estimate the skeletal joints with higher precision in millimeter. Joint variances are obtained from the maximum likelihood optimization which is used as local human part feature representation and is directly utilized for 3D human point clouds reconstruction. The proposed network achieved an average of 32.17mm joint distance deviation against ground truth joints and 0.62×10^{-3} average Chamfer distance on reconstruction fidelity.

The synthetic model is a skeletal joint-based regressive model that reconstructs synthetic 3D human from partial point cloud. The network first estimates the skeletal joints components of input partial point cloud and a regressive 3D human reconstructs a 3D human model based on the estimated skeletal joints. A superposed MLP comprising matrix vectorization and parameters superposition in the skeletal joints encoder is adopted to improve both memory footprint and processing efficiency. In addition, a two-mode model optimization strategy consisting network training on synthesized data and fine tuning on real-world data is proposed. This allows generation of simulated partial point cloud for network training that would be resource expensive if acquired in real-world. The network is weakly supervised guided by ground truth joints and maximum likelihood of ground truth point cloud. The network achieved an average of 24.83mm joint distance deviation against ground truth joints and 0.84×10^{-3} average Chamfer distance on reconstruction fidelity.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

The use of three dimensional (3D) information can be widely found in the application of 3D modeling, autonomous robotics navigation, semantic scene understanding. Particularly, 3D modeling plays an important role to aid the modeling of real-world object into a digitized 3D model to better understand the properties and characteristics of the object. The acquisition of 3D data can be achieved using three methods, i.e. time-of-flight, structured-light and stereo vision. In brief, 3D data representation can be categorized into RGB-D, voxel and point cloud that can better suit for specified applications. In the current research trend, point cloud is favored over other representations due to its raw and simple form of representation in XYZ axes, and contains richer geometric information. Nonetheless, there are still challenges in directly processing raw point cloud to perform 3D modeling such as un-ordered structure and partial absence of point cloud.

The latest 3D point cloud reconstruction techniques in the process of 3D modeling are revealed and it can be divided into three categories, i.e. statistical models, discriminative learning models and generative learning models. Conventional 3D point cloud reconstruction typically implements statistical and discriminative learning models in establishing correspondences between multiple viewpoint to estimate the required transformation. Essentially statistical models analyse the input data and extract feature descriptor for maximum likelihood matching using mathematical derivation of features. In contrast to statistical models, discriminative learning models use supervised learning approach to establish local correspondence driven by labeled data learning. These models allow higher capability of task-oriented learning, novel data generalization and more robust to artifacts. In the advent of generative learning models, 3D data are processed directly by the network, resulting improved efficacy of 3D reconstruction process without the need of multiple viewpoint inputs. It is observed generative learning models would be the active research trend in 3D reconstruction techniques due to its advantages, hence generative learning model is adopted as the backbone in this research work.

A generative learning model begins in 3D data representation learning to gather the representative features of input 3D data. A novel point cloud representation learning technique is designed using proposed Point cloud Neural Density Estimator (PNDE) derived from Gaussian mixture model. The network is architectural designed to adopts the concept of neural density estimation using maximum log-likelihood of multivariate Gaussian mixture model as an objective function for end-to-end network training. The working principle of PNDE is to estimate the density of point cloud by encoding a global latent density and subsequently infers the local densities. The output of PNDE is a set of density parameters (means and variances) and they can be directly used for classification and generative task such as reconstruction tasks. From experimentation, the estimated densities are evaluated on point cloud classification task. The results show promising performance in point cloud representation using density parameters, achieving comparable results to the state-of-the-art methods with overall accuracy of 93.67% in ModelNet10, 88.74% in ModelNet40 and 94.6% in ShapNetCore13. The proposed PNDE is a scalable network by adjusting the number of output densities, hence the resolution of representation can be flexibly adjusted according to application and computation resources. In addition, a network compression strategy is implemented to reduce up to 8$\times$ smaller the original network parameter size, while retaining approximately 1.5% degradation in classification accuracy. On the other hand, the proposed PNDE is qualitatively evaluated on the property of equivariant, where the transformation imposed in the input is conveyed to the estimated densities.

Intuitively, the use of density parameters can be extended to perform point cloud reconstruction by adopting a generative sampling process using a decoder network. To achieve this, a novel partial point cloud reconstruction model using Gaussian components (density parameters) as compressed representation, Gaussian Point cloud Autoencoder (GPAE) is designed to reconstruct a complete 3D points point from single viewpoint partial point cloud. In this network, patch-based decoder is adopted to perform generative sampling by taking input of the Gaussian components. Subsequently, GPAE is trained end-to-end using reconstruction loss as the objective function. From experimentation, the proposed GPAEs outperformed several existing works with lowest average CD of 0.56×10^{-3} despite not having point cloud refinement process and 97% reconstruction accuracy is achieved within 5% deviation relative to ground truth. In addition, the network is robust on highly sparse input up to 50% missing points with a sustained degradation of average 8% in reconstruction quality. Further, thanks to the network compression strategy proposed in PNDE, GPAEs have significantly smaller network size with up to 50× smaller compared to others. Overall, the propoesd GPAEs showed generalized point cloud shapes learning as evaluated using metrics such as Chamfer Distance (CD) and Earth Mover Distance (EMD) across 3D objects. Using the designed reconstruction technique, edge implementation can be greatly benefited due to smaller network size and with relatively good performance in coarse reconstruction. While, in the case where refined output is demanded, it can be achieved by reiterate the forward propagation and merging process.

By using GPAE as a generative learning model backbone, a novel part-to-whole learning network is designed for point cloud learning and reconstruction. In order to enforce the part-to-whole learning mechanism, a point cloud part segmentation network, Part Sampler network is designed using PNDE, and a capsule network is adopted to learn the reasoning of an object's existence using agreement on parts voting through dynamic routing algorithm. For generative reconstruction purpose, decoder network inspired from GPAE is attached to the object capsules for point cloud reconstruction, while acting as an addition guidance to optimize the object capsules. As the proposed network learns from segmented parts, it requires significantly lower latent dimension in part capsule compared to existing works that use higher latent dimension and rely on large network parameters to regress to a generalized part features. From experimentation, the proposed network is evaluated on classification tasks, achieving testing classification accuracy of 93.56% in ModelNet10, 88.70% in ModelNet40 and 94.71% in ShapeNetCore13. Moreover, the proposed network inherited as a equivariant network benefited from presence of equivariance in Part Sampler and capsule network. Thus, the proposed network achieved test accuracy of 86.67% in ModelNet10, 79.34% in ModelNet40 and 87.83% in ShapeNet13, where the input point clouds are imposed with rotation perturbation. In qualitative results, the proposed network is shown to fully reconstruct output that is equivariant to input point cloud with imposed rotation perturbation. Furthermore, the reconstructed output of the proposed network is quantitatively evaluated using ICP algorithm and achieved average $\pm 2\,\text{deg}$ discrepancies in rotation. Hence, with the advantages in the part-to-whole learning network, real-time applications that are dynamically moving are able to perform with higher consistency.

Lastly, a real-world application in 3D human modeling is demonstrated using proposed skeletal joint-based generative learning model to reconstruct 3D human in synthetic and non-synthetic representation. Two 3D human reconstruction models are proposed: 1) generative non-synthetic model, and 2) regressive synthetic model to obtain complete representation of 3D human by taking input sparse partial point cloud acquired from single viewpoint depth camera. The two 3D human reconstruction models share similar initial stage by estimating the skeletal components, i.e. skeletal joints and variances, using a proposed skeletal joints encoder inspired from PNDE. Subsequently, non-synthetic 3D human in point cloud is generated by generative sampling on the estimated skeletal components using decoder inspired from GPAE; while synthetic 3D human model is generated by regressing the model through inverse kinematic. From experiment, the generative model achieved an average of $32.17mm$ joint distance deviation against ground truth joints and 0.62×10^{-3} average Chamfer distance on reconstruction fidelity. On the other hand, the regressive model achieved an average of $24.83mm$ joint distance deviation against ground truth joints and 0.84×10^{-3} average Chamfer distance on reconstruction fidelity. Essentially, the improvement on the regressive model is gained by a two-mode model optimization strategy consisting network training on synthesized data and fine tuning on real-world data. Nonetheless, the proposed skeletal joint-based generative learning models can provide crucial information of 3D human with higher accuracy in estimated skeletal joints, non-synthetic point cloud with less artifacts and noises, and synthetic skinned human. Therefore, potential applications that can be benefited from the advancement of the models are sport science where skeletal joints are important for posture analysis and 3D human modeling in film industry.

7.2 Future Work

In the future progress, several improvements can be incrementally developed to overcome the limitations of proposed networks in this research. Firstly, PNDE relies on number of MLPs that is linearly proportional to the number of output densities.

The current design of PNDE implemented matrix factorization and vector superposition to reduce the network parameter as well as improve the network propagation, which allowed the network to operate in real-time inference in low number of output densities M. However, the network may encounter scalability issue when M is set to a significantly large number. This is particularly significant in application with large dataset such as large scene and open world environment. To overcome this, hierarchical recursive PNDEs can be implemented to hierarchically estimate lower rank densities from a higher rank density.

Next, learning in partial point cloud posses high degree of difficulty due to the missing information in the data. Intuitively, by treating each sample of partial point cloud as variations in viewpoint, learning of partial point cloud can be viewed as learning invariant representation of distorted input samples. Therefore, from within the distorted samples, exist trivial and redundant constant solutions that are embedded in each distorted input sample. An improved self-supervised learning using barlow twins can be implemented in GPAE training to efficiently learn the invariant representation of partial point cloud through redundancy reduction. Subsequently, the designed part-to-whole learning network is implemented using vector features embed both pose invariant geometric features and pose features. Hence, the pose features are constantly entangled in the geometric features, preventing the obtainment of pose features. The pose features is particularly important in understanding the transformation that is imposed in the input for task such as viewpoint estimation. In the future development, pose features can be independently incorporated into the capsule network to obtain both geometric features and pose features. This can also ensure the disentanglement of pose in the geometric features and thus allowing independent utilization of the pose and geometric features.

Lastly, several incremental works can be implemented in the future to improve the 3D human modeling network model. Firstly, a bijective function distance metric such as Earth Mover Distance can be implemented for guiding the network to produce higher fidelity output. This is due to current implemented Chamfer distance that does not take into consideration of surface uniformity and density, therefore the completed point clouds may create visual artifacts. Further, a pose-based encoder can be developed to output joints rotation information to directly drive the regressive 3D human model. This is due to Vposer, which is an inverse kinematic based 3D human regression method, hence the regressed 3D human model may present a slightly distorted pose and therefore affects the accuracy of reconstructed human shape.

References

[1] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with microsoft kinect sensor: A review," *IEEE transactions on cybernetics*, vol. 43, no. 5, pp. 1318–1334, 2013.

[2] M. Draelos, Q. Qiu, A. Bronstein, and G. Sapiro, "Intel realsense= real low cost gaze," in *2015 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2015, pp. 2520–2524.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[4] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[5] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[6] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1520–1528.

[7] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.

[8] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[9] P. Kim, J. Chen, and Y. K. Cho, "Slam-driven robotic mapping and registration of 3d point clouds," *Automation in Construction*, vol. 89, pp. 38–48, 2018.

[10] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3dmatch: Learning local geometric descriptors from rgb-d reconstructions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1802–1811.

[11] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1915–1929, 2012.

[12] S. Srivastava and B. Lall, "Deeppoint3d: Learning discriminative local descriptors using deep metric learning on 3d point clouds," *Pattern Recognition Letters*, 2019.

[13] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4558–4567.

[14] Y. Cheng, B. Yang, B. Wang, W. Yan, and R. T. Tan, "Occlusion-aware networks for 3d human pose estimation in video," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 723–732.

[15] G. Pavlakos, X. Zhou, K. G. Derpanis, and K. Daniilidis, "Coarse-to-fine volumetric prediction for single-image 3d human pose," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7025–7034.

[16] H. Jiang, J. Cai, and J. Zheng, "Skeleton-aware 3d human shape reconstruction from point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5431–5441.

[17] Y. Zhou, H. Dong, and A. El Saddik, "Learning to estimate 3d human pose from point cloud," *IEEE Sensors Journal*, vol. 20, no. 20, pp. 12 334–12 342, 2020.

[18] M. Zollhöfer, "Commodity rgb-d sensors: Data acquisition," in *RGB-D Image Analysis and Processing.* Springer, 2019, pp. 3–13.

[19] T. Xu, D. An, Y. Jia, and Y. Yue, "A review: Point cloud-based 3d human joints estimation," *Sensors*, vol. 21, no. 5, p. 1684, 2021.

[20] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3d point clouds: A survey," *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[21] C. Jing, J. Potgieter, F. Noble, and R. Wang, "A comparison and analysis of rgb-d cameras' depth performance for robotics application," in *2017 24th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*. IEEE, 2017, pp. 1–6.

[22] E. Ahmed, A. Saint, A. E. R. Shabayek, K. Cherenkova, R. Das, G. Gusev, D. Aouada, and B. Ottersten, "A survey on deep learning advances on different 3d data representations," *arXiv preprint arXiv:1808.01462*, 2018.

[23] M. Zollhöfer, P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb, "State of the art on 3d reconstruction with rgb-d cameras," in *Computer Graphics Forum*, vol. 37, no. 2. Wiley Online Library, 2018, pp. 625–652.

[24] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in *Proceedings of the 15th ACM international conference on Multimedia*. ACM, 2007, pp. 357–360.

[25] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 3212–3217.

[26] S. Salti, F. Tombari, and L. Di Stefano, "Shot: Unique signatures of histograms for surface and texture description," *Computer Vision and Image Understanding*, vol. 125, pp. 251–264, 2014.

[27] Q. Yu, J. Liang, J. Xiao, H. Lu, and Z. Zheng, "A novel perspective invariant feature transform for rgb-d images," *Computer Vision and Image Understanding*, vol. 167, pp. 109–120, 2018.

[28] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.

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

[30] G. Riegler, A. Osman Ulusoy, and A. Geiger, "Octnet: Learning deep 3d representations at high resolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3577–3586.

[31] H. Huang, E. Kalogerakis, S. Chaudhuri, D. Ceylan, V. G. Kim, and E. Yumer, "Learning local shape descriptors from part correspondences with multiview convolutional networks," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 1, p. 6, 2018.

[32] S. Srivastava and B. Lall, "Deeppoint3d: Learning discriminative local descriptors using deep metric learning on 3d point clouds," *Pattern Recognition Letters*, 2019.

[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[34] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "Pcn: Point completion network," in *2018 International Conference on 3D Vision (3DV)*. IEEE, 2018, pp. 728–737.

[35] M. Liu, L. Sheng, S. Yang, J. Shao, and S.-M. Hu, "Morphing and sampling network for dense point cloud completion," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 11 596–11 603.

[36] Y. Yang, C. Feng, Y. Shen, and D. Tian, "Foldingnet: Point cloud auto-encoder via deep grid deformation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 206–215.

[37] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3d point clouds," in *International conference on machine learning*, 2018, pp. 40–49.

[38] H. Xie, H. Yao, S. Zhou, J. Mao, S. Zhang, and W. Sun, "Grnet: gridding residual network for dense point cloud completion," in *European Conference on Computer Vision*. Springer, 2020, pp. 365–381.

[39] W. Yuan, D. Held, C. Mertz, and M. Hebert, "Iterative transformer network for 3d point cloud," *arXiv preprint arXiv:1811.11209*, 2018.

[40] G. E. Hinton, A. Krizhevsky, and S. D. Wang, "Transforming auto-encoders," in *International Conference on Artificial Neural Networks.* Springer, 2011, pp. 44–51.

[41] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in neural information processing systems*, 2017, pp. 3856–3866.

[42] N. Srivastava, H. Goh, and R. Salakhutdinov, "Geometric capsule autoencoders for 3d point clouds," *arXiv preprint arXiv:1912.03310*, 2019.

[43] M. Dou, S. Khamis, Y. Degtyarev, P. Davidson, S. R. Fanello, A. Kowdle, S. O. Escolano, C. Rhemann, D. Kim, J. Taylor *et al.*, "Fusion4d: Real-time performance capture of challenging scenes," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 114, 2016.

[44] L. Xie, X. Zhang, Y. Xu, Y. Shang, and Q. Yu, "Skeletonfusion: Reconstruction and tracking of human body in real-time," *Optics and Lasers in Engineering*, vol. 110, pp. 80–88, 2018.

[45] J. Li, J. Yang, Z. Xu, and J. Peng, "Computer-assisted hand rehabilitation assessment using an optical motion capture system," in *2012 International Conference on Image Analysis and Signal Processing.* IEEE, 2012, pp. 1–5.

[46] J. Guna, G. Jakus, M. Pogačnik, S. Tomažič, and J. Sodnik, "An analysis of the precision and reliability of the leap motion sensor and its suitability for static and dynamic tracking," *Sensors*, vol. 14, no. 2, pp. 3702–3720, 2014.

[47] B. Fei, W. Yang, W. Chen, Z. Li, Y. Li, T. Ma, X. Hu, and L. Ma, "Comprehensive review of deep learning-based 3d point clouds completion processing and analysis," *arXiv preprint arXiv:2203.03311*, 2022.

[48] Y. Zhao, T. Birdal, J. E. Lenssen, E. Menegatti, L. Guibas, and F. Tombari, "Quaternion equivariant capsule networks for 3d point clouds," in *European Conference on Computer Vision.* Springer, 2020, pp. 1–19.

[49] A. Cheraghian and L. Petersson, “3dcapsule: Extending the capsule architecture to classify 3d point clouds,” in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV).* IEEE, 2019, pp. 1194–1202.

[50] M. Berger, A. Tagliasacchi, L. Seversky, P. Alliez, J. Levine, A. Sharf, and C. Silva, “State of the art in surface reconstruction from point clouds,” in *Eurographics 2014-State of the Art Reports*, vol. 1, no. 1, 2014, pp. 161–185.

[51] Y. Wang, D. J. Tan, N. Navab, and F. Tombari, “Softpoolnet: Shape descriptor for point cloud completion and classification,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16.* Springer, 2020, pp. 70–85.

[52] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, “Dynamic graph cnn for learning on point clouds,” *Acm Transactions On Graphics (tog)*, vol. 38, no. 5, pp. 1–12, 2019.

[53] R. Cai, G. Yang, H. Averbuch-Elor, Z. Hao, S. Belongie, N. Snavely, and B. Hariharan, “Learning gradient fields for shape generation,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16.* Springer, 2020, pp. 364–381.

[54] A. Hertz, R. Hanocka, R. Giryes, and D. Cohen-Or, “Pointgmm: A neural gmm network for point clouds,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 054–12 063.

[55] L. Pan, X. Chen, Z. Cai, J. Zhang, H. Zhao, S. Yi, and Z. Liu, “Variational relational point completion network,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8524–8533.

[56] C. Nash and C. K. Williams, “The shape variational autoencoder: A deep generative model of part-segmented 3d objects,” in *Computer Graphics Forum*, vol. 36. Wiley Online Library, 2017, pp. 1–12.

[57] G. Yang, X. Huang, Z. Hao, M.-Y. Liu, S. Belongie, and B. Hariharan, “Pointflow: 3d point cloud generation with continuous normalizing flows,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 4541–4550.

[58] L. Zhou, S. Zhu, Z. Luo, T. Shen, R. Zhang, M. Zhen, T. Fang, and L. Quan, "Learning and matching multi-view descriptors for registration of point clouds," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 505–522.

[59] R. Yao, G. Lin, S. Xia, J. Zhao, and Y. Zhou, "Video object segmentation and tracking: A survey," *arXiv preprint arXiv:1904.09172*, 2019.

[60] X. Lu, W. Wang, J. Shen, Y.-W. Tai, D. J. Crandall, and S. C. Hoi, "Learning video object segmentation from unlabeled videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8960–8970.

[61] X. Lu, C. Ma, B. Ni, and X. Yang, "Adaptive region proposal with channel regularization for robust object tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.

[62] G. Ciaparrone, F. L. Sánchez, S. Tabik, L. Troiano, R. Tagliaferri, and F. Herrera, "Deep learning in video multi-object tracking: A survey," *Neurocomputing*, vol. 381, pp. 61–88, 2020.

[63] H. Altwaijry, A. Veit, S. J. Belongie, and C. Tech, "Learning to detect and match keypoints with deep architectures." in *BMVC*, 2016.

[64] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[65] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *IEEE Access*, vol. 7, pp. 128 837–128 868, 2019.

[66] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view cnns for object classification on 3d data," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5648–5656.

[67] R. Klokov and V. Lempitsky, "Escape from cells: Deep kd-networks for the recognition of 3d point cloud models," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 863–872.

[68] E. Grilli, F. Menna, and F. Remondino, “A review of point clouds segmentation and classification algorithms,” *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, p. 339, 2017.

[69] H. RaviPrakash, S. M. Anwar, and U. Bagci, “Variational capsule encoder,” *arXiv preprint arXiv:2010.09102*, 2020.

[70] G. E. Hinton, S. Sabour, and N. Frosst, “Matrix capsules with em routing,” in *International conference on learning representations*, 2018.

[71] Z. Wang and F. Lu, “Voxsegnet: Volumetric cnns for semantic part segmentation of 3d shapes,” *IEEE transactions on visualization and computer graphics*, vol. 26, no. 9, pp. 2919–2930, 2019.

[72] C. Wang, M. Cheng, F. Sohel, M. Bennamoun, and J. Li, “Normalnet: A voxel-based cnn for 3d object classification and retrieval,” *Neurocomputing*, vol. 323, pp. 139–147, 2019.

[73] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.

[74] A. Xiao, J. Huang, D. Guan, and S. Lu, “Unsupervised representation learning for point clouds: A survey,” *arXiv preprint arXiv:2202.13589*, 2022.

[75] L. Nanni, S. Ghidoni, and S. Brahnam, “Handcrafted vs. non-handcrafted features for computer vision classification,” *Pattern Recognition*, vol. 71, pp. 158–172, 2017.

[76] M. S. Patel, N. Patel, and M. S. Holia, “Feature based multi-view image registration using surf,” in *2015 International Symposium on Advanced Computing and Communication (ISACC)*. IEEE, 2015, pp. 213–218.

[77] Z. Li and C. Chen, “An improved adaptive threshold brisk feature matching algorithm based on surf,” in *2018 Chinese Automation Congress (CAC)*. IEEE, 2018, pp. 2928–2932.

[78] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf,” in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.

[79] J. Ligon, D. Bein, P. Ly, and B. Onesto, "3d point cloud processing using spin images for object detection," in *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 2018, pp. 731–736.

[80] M. Imanullah, E. M. Yuniarno, and S. Sumpeno, "Sift and icp in multi-view based point clouds registration for indoor and outdoor scene reconstruction," in *2019 International Seminar on Intelligent Technology and Its Applications (ISITIA)*. IEEE, 2019, pp. 288–293.

[81] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, "Aligning point cloud views using persistent feature histograms," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 3384–3391.

[82] L. Shapira, S. Shalom, A. Shamir, D. Cohen-Or, and H. Zhang, "Contextual part analogies in 3d objects," *International Journal of Computer Vision*, vol. 89, no. 2-3, pp. 309–326, 2010.

[83] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan, "Rotational projection statistics for 3d local surface description and object recognition," *International journal of computer vision*, vol. 105, no. 1, pp. 63–86, 2013.

[84] V. G. Kim, W. Li, N. J. Mitra, S. Chaudhuri, S. DiVerdi, and T. Funkhouser, "Learning part-based templates from large collections of 3d shapes," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, p. 70, 2013.

[85] H. H. Tan and K. H. Lim, "Vanishing gradient mitigation with deep learning neural network optimization," in *2019 7th International Conference on Smart Computing & Communications (ICSCC)*. IEEE, 2019, pp. 1–4.

[86] Y. Fang, J. Xie, G. Dai, M. Wang, F. Zhu, T. Xu, and E. Wong, "3d deep shape descriptor," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2319–2328.

[87] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta, "Learning a predictable and generative vector representation for objects," in *European Conference on Computer Vision*. Springer, 2016, pp. 484–499.

[88] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling," in *Advances in neural information processing systems*, 2016, pp. 82–90.

[89] A. Dai, C. Ruizhongtai Qi, and M. Nießner, “Shape completion using 3d-encoder-predictor cnns and shape synthesis,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5868–5877.

[90] A. Kar, C. Häne, and J. Malik, “Learning a multi-view stereo machine,” *arXiv preprint arXiv:1708.05375*, 2017.

[91] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, “3d-r2n2: A unified approach for single and multi-view 3d object reconstruction,” in *European conference on computer vision.* Springer, 2016, pp. 628–644.

[92] M. Tatarchenko, A. Dosovitskiy, and T. Brox, “Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2088–2096.

[93] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, “O-cnn: Octree-based convolutional neural networks for 3d shape analysis,” *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–11, 2017.

[94] M. Khoury, Q.-Y. Zhou, and V. Koltun, “Learning compact geometric features,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 153–161.

[95] D. Valsesia, G. Fracastoro, and E. Magli, “Learning localized generative models for 3d point clouds via graph convolution,” in *International conference on learning representations*, 2018.

[96] D. W. Shu, S. W. Park, and J. Kwon, “3d point cloud generative adversarial network based on tree structured graph convolutions,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3859–3868.

[97] G. Koch, R. Zemel, and R. Salakhutdinov, “Siamese neural networks for one-shot image recognition,” in *ICML deep learning workshop*, vol. 2. Lille, 2015.

[98] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler, “Learning by tracking: Siamese cnn for robust target association,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 33–40.

[99] A. Zhu, J. Yang, C. Zhao, K. Xian, Z. Cao, and X. Li, “Lrf-net: Learning local reference frames for 3d local shape description and matching,” *arXiv preprint arXiv:2001.07832*, 2020.

[100] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, “A papier-mâché approach to learning 3d surface generation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 216–224.

[101] X. Wang, M. H. Ang Jr, and G. H. Lee, “Cascaded refinement network for point cloud completion,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 790–799.

[102] L. P. Tchapmi, V. Kosaraju, H. Rezatofighi, I. Reid, and S. Savarese, “Topnet: Structural point cloud decoder,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[103] X. Wen, T. Li, Z. Han, and Y.-S. Liu, “Point cloud completion by skip-attention network with hierarchical folding,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1939–1948.

[104] Y. Zhao, T. Birdal, H. Deng, and F. Tombari, “3d point capsule networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 1009–1018.

[105] X. Wen, Z. Han, X. Liu, and Y.-S. Liu, “Point2spatialcapsule: Aggregating features and spatial relationships of local regions on point clouds using spatial-aware capsules,” *arXiv preprint arXiv:1908.11026*, 2019.

[106] A. Handa, T. Whelan, J. McDonald, and A. Davison, “A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM,” in *IEEE Intl. Conf. on Robotics and Automation, ICRA*, Hong Kong, China, May 2014.

[107] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys, “SEMANTIC3D.NET: A new large-scale point cloud classification benchmark,” in *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. IV-1-W1, 2017, pp. 91–98.

[108] S. Azimi and T. K. Gandhi, “Performance comparison of 3d correspondence grouping algorithm for 3d plant point clouds,” *arXiv preprint arXiv:1909.00866*, 2019.

[109] H. Fan, H. Su, and L. J. Guibas, “A point set generation network for 3d object reconstruction from a single image,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 605–613.

[110] A. McGregor and D. Stubbs, “Sketching earth-mover distance on graph metrics,” in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 274–286.

[111] S. Srivastava, G. Sharma, and B. Lall, “Large scale novel object discovery in 3d,” in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV).* IEEE, 2018, pp. 179–188.

[112] N. Blanchard, K. Skinner, A. Kemp, W. J. Scheirer, and P. J. Flynn, “"keep me in, coach!": A computer vision perspective on assessing acl injury risk in female athletes,” *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1366–1374, 2019.

[113] C. Z. Tay, K. H. Lim, and J. T. S. Phang, “Markerless gait estimation and tracking for postural assessment,” *Multimedia Tools and Applications*, Feb 2022. [Online]. Available: https://doi.org/10.1007/s11042-022-12026-8

[114] S. Li and A. B. Chan, “3d human pose estimation from monocular images with deep convolutional neural network,” in *Asian Conference on Computer Vision.* Springer, 2014, pp. 332–347.

[115] X. Zhou, Q. Huang, X. Sun, X. Xue, and Y. Wei, “Towards 3d human pose estimation in the wild: a weakly-supervised approach,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 398–407.

[116] B. Tekin, I. Katircioglu, M. Salzmann, V. Lepetit, and P. Fua, “Structured prediction of 3d human pose with deep neural networks,” *arXiv preprint arXiv:1605.05180*, 2016.

[117] C. Zheng, W. Wu, T. Yang, S. Zhu, C. Chen, R. Liu, J. Shen, N. Kehtarnavaz, and M. Shah, “Deep learning-based human pose estimation: A survey,” *arXiv preprint arXiv:2012.13392*, 2020.

[118] G. Pavlakos, X. Zhou, and K. Daniilidis, “Ordinal depth supervision for 3d human pose estimation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7307–7316.

[119] X. Sun, J. Shang, S. Liang, and Y. Wei, "Compositional human pose regression," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2602–2611.

[120] L. Zhao, X. Peng, Y. Tian, M. Kapadia, and D. N. Metaxas, "Semantic graph convolutional networks for 3d human pose regression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3425–3435.

[121] J. Xu, Z. Yu, B. Ni, J. Yang, X. Yang, and W. Zhang, "Deep kinematics analysis for monocular 3d human pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 899–908.

[122] F. Bogo, J. Romero, M. Loper, and M. J. Black, "Faust: Dataset and evaluation for 3d mesh registration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3794–3801.

[123] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "Smpl: A skinned multi-person linear model," *ACM transactions on graphics (TOG)*, vol. 34, no. 6, pp. 1–16, 2015.

[124] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black, "Keep it smpl: Automatic estimation of 3d human pose and shape from a single image," in *European conference on computer vision.* Springer, 2016, pp. 561–578.

[125] G. Pavlakos, V. Choutas, N. Ghorbani, T. Bolkart, A. A. A. Osman, D. Tzionas, and M. J. Black, "Expressive body capture: 3d hands, face, and body from a single image," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[126] A. A. Osman, T. Bolkart, and M. J. Black, "Star: Sparse trained articulated human body regressor," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16.* Springer, 2020, pp. 598–613.

[127] O. Litany, A. Bronstein, M. Bronstein, and A. Makadia, "Deformable shape completion with graph convolutional autoencoders," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1886–1895.

[128] Q. Tan, L. Gao, Y.-K. Lai, and S. Xia, “Variational autoencoders for deforming 3d mesh models,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5841–5850.

[129] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, “Pixel2mesh: Generating 3d mesh models from single rgb images,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 52–67.

[130] K. Greff, S. Van Steenkiste, and J. Schmidhuber, “Neural expectation maximization,” in *Advances in Neural Information Processing Systems*, 2017, pp. 6691–6701.

[131] G. Papamakarios, “Neural density estimation and likelihood-free inference,” *arXiv preprint arXiv:1910.13233*, 2019.

[132] D. Shin, C. C. Fowlkes, and D. Hoiem, “Pixels, voxels, and views: A study of shape representations for single view 3d object shape prediction,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3061–3069.

[133] M. Simonovsky and N. Komodakis, “Dynamic edge-conditioned filters in convolutional neural networks on graphs,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3693–3702.

[134] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, “Pointcnn: Convolution on {X}-transformed points,” *arXiv preprint arXiv:1801.07791*, 2018.

[135] T. S. Cohen, M. Weiler, B. Kicanaoglu, and M. Welling, “Gauge equivariant convolutional networks and the icosahedral cnn,” *arXiv preprint arXiv:1902.04615*, 2019.

[136] H. Lei, N. Akhtar, and A. Mian, “Spherical kernel for efficient graph convolution on 3d point clouds,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[137] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, “Adversarial autoencoders,” 2016.

[138] M. Zamorski, M. Zięba, P. Klukowski, R. Nowak, K. Kurach, W. Stokowiec, and T. Trzciński, “Adversarial autoencoders for compact representations of 3d point clouds,” *Computer Vision and Image Understanding*, vol. 193, p. 102921, 2020.

[139] C.-H. Chen, F. Song, F.-J. Hwang, and L. Wu, "A probability density function generator based on neural networks," *Physica A: Statistical Mechanics and its Applications*, vol. 541, p. 123344, 2020.

[140] P. Chilinski and R. Silva, "Neural likelihoods via cumulative distribution functions," in *Conference on Uncertainty in Artificial Intelligence.* PMLR, 2020, pp. 420–429.

[141] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[142] S. Albanie, "Euclidean distance matrix trick," *Retrieved from Visual Geometry Group, University of Oxford*, 2019.

[143] Y. Wen, D. Tran, and J. Ba, "Batchensemble: an alternative approach to efficient ensemble and lifelong learning," *arXiv preprint arXiv:2002.06715*, 2020.

[144] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.

[145] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[146] J. E. Lenssen, M. Fey, and P. Libuschewski, "Group equivariant capsule networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 8844–8853.

[147] R. Kondor and S. Trivedi, "On the generalization of equivariance and convolution in neural networks to the action of compact groups," in *International Conference on Machine Learning.* PMLR, 2018, pp. 2747–2755.

[148] R. Rocci, S. A. Gattone, and R. Di Mari, "A data driven equivariant approach to constrained gaussian mixture modeling," *Advances in Data Analysis and Classification*, vol. 12, no. 2, pp. 235–260, 2018.

[149] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view cnns for object classification on 3d data," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5648–5656.

[150] S.-S. Yu, S.-W. Chu, C.-M. Wang, Y.-K. Chan, and T.-C. Chang, "Two improved k-means algorithms," *Applied Soft Computing*, vol. 68, pp. 747–755, 2018.

[151] W. Huang and Y. Chen, "The multiset em algorithm," *Statistics & Probability Letters*, vol. 126, pp. 41–48, 2017.

[152] B. Cheung, A. Terekhov, Y. Chen, P. Agrawal, and B. Olshausen, "Superposition of many models into one," in *Advances in Neural Information Processing Systems*, 2019, pp. 10 868–10 877.

[153] M. Tatarchenko, S. R. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox, "What do single-view 3d reconstruction networks learn?" in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3405–3414.

[154] M. Sung, H. Su, R. Yu, and L. Guibas, "Deep functional dictionaries: Learning consistent semantic structures on 3d models from functions," *arXiv preprint arXiv:1805.09957*, 2018.

[155] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[156] R. Li, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "Pu-gan: a point cloud upsampling adversarial network," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7203–7212.

[157] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis, "A survey on 3d object detection methods for autonomous driving applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3782–3795, 2019.

[158] Q. Hu, B. Yang, S. Khalid, W. Xiao, N. Trigoni, and A. Markham, "Towards semantic segmentation of urban-scale 3d point clouds: A dataset, benchmarks and challenges," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4977–4987.

[159] L. Yi, B. Gong, and T. Funkhouser, "Complete & label: A domain adaptation approach to semantic segmentation of lidar point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 363–15 373.

[160] Z. Huang, Y. Yu, J. Xu, F. Ni, and X. Le, "Pf-net: Point fractal network for 3d point cloud completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7662–7670.

[161] M. Xu, R. Ding, H. Zhao, and X. Qi, "Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3173–3182.

[162] K. Zhang, M. Hao, J. Wang, C. W. de Silva, and C. Fu, "Linked dynamic graph cnn: Learning on point cloud via linking hierarchical features," *arXiv preprint arXiv:1904.10014*, 2019.

[163] K. Wang, K. Chen, and K. Jia, "Deep cascade generation on point sets." in *IJCAI*, vol. 2, 2019, p. 4.

[164] K. Hassani and M. Haley, "Unsupervised multi-task feature learning on point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 8160–8171.

[165] D. Liu, C. Chen, C. Xu, Q. Cai, L. Chu, and R. C. Qiu, "A self contour-based rotation and translation-invariant transformation for point clouds recognition," *arXiv preprint arXiv:2009.06903*, 2020.

[166] H. Deng, T. Birdal, and S. Ilic, "Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 602–618.

[167] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley, "Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds," *arXiv preprint arXiv:1802.08219*, 2018.

[168] R. Spezialetti, S. Salti, and L. D. Stefano, "Learning an effective equivariant 3d descriptor without supervision," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6401–6410.

[169] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, "Barlow twins: Self-supervised learning via redundancy reduction," *arXiv preprint arXiv:2103.03230*, 2021.

[170] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the effective receptive field in deep convolutional neural networks," *Advances in neural information processing systems*, vol. 29, 2016.

[171] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. International Society for Optics and Photonics, 1992, pp. 586–606.

[172] L. Wang, Z. Ding, Z. Tao, Y. Liu, and Y. Fu, "Generative multi-view human action recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6212–6221.

[173] M. K. Bekele, R. Pierdicca, E. Frontoni, E. S. Malinverni, and J. Gain, "A survey of augmented, virtual, and mixed reality for cultural heritage," *Journal on Computing and Cultural Heritage (JOCCH)*, vol. 11, no. 2, pp. 1–36, 2018.

[174] J. Wang, K. Qiu, H. Peng, J. Fu, and J. Zhu, "Ai coach: Deep human pose estimation and analysis for personalized athletic training assistance," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 374–382.

[175] G. Fahim, K. Amin, and S. Zarif, "Single-view 3d reconstruction: A survey of deep learning methods," *Computers & Graphics*, vol. 94, pp. 164–190, 2021.

[176] B. L. Bhatnagar, C. Sminchisescu, C. Theobalt, and G. Pons-Moll, "Combining implicit function learning and parametric models for 3d human reconstruction," *arXiv preprint arXiv:2007.11432*, 2020.

[177] M. Kresović and T. D. Nguyen, "Bottom-up approaches for multi-person pose estimation and it's applications: A brief review," *arXiv preprint arXiv:2112.11834*, 2021.

[178] Z.-Q. Cheng, Y. Chen, R. R. Martin, T. Wu, and Z. Song, "Parametric modeling of 3d human body shape—a survey," *Computers & Graphics*, vol. 71, pp. 88–100, 2018.

[179] G. Pavlakos, L. Zhu, X. Zhou, and K. Daniilidis, "Learning to estimate 3d human pose and shape from a single color image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 459–468.

[180] I. Ramirez, A. Cuesta-Infante, E. Schiavi, and J. J. Pantrigo, “Bayesian capsule networks for 3d human pose estimation from single 2d images,” *Neurocomputing*, vol. 379, pp. 64–73, 2020.

[181] G. Varol, D. Ceylan, B. Russell, J. Yang, E. Yumer, I. Laptev, and C. Schmid, “Bodynet: Volumetric inference of 3d human body shapes,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 20–36.

[182] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, “Pct: Point cloud transformer,” *arXiv preprint arXiv:2012.09688*, 2020.

[183] G. Pavlakos, V. Choutas, N. Ghorbani, T. Bolkart, A. A. Osman, D. Tzionas, and M. J. Black, “Expressive body capture: 3d hands, face, and body from a single image,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 975–10 985.

[184] M. Xu, W. Dai, Y. Shen, and H. Xiong, “Msgcnn: Multi-scale graph convolutional neural network for point cloud segmentation,” in *2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM)*. IEEE, 2019, pp. 118–127.

[185] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, “Berkeley mhad: A comprehensive multimodal human action database,” in *2013 IEEE Workshop on Applications of Computer Vision (WACV)*. IEEE, 2013, pp. 53–60.