Point Set Voting for Partial Point Cloud Analysis

Junming Zhang, Weijia Chen, Yuping Wang

Ram Vasudevan, Matthew Johnson-Roberson

University of Michigan
{junming, weijiac, ypw, ramv, mattjr}@umich.edu

Abstract

The continual improvement of 3D sensors has driven the development of algorithms to perform point cloud analysis. In fact, techniques for point cloud classification and segmentation have in recent years achieved incredible performance driven in part by leveraging large synthetic datasets. Unfortunately these same state-of-the-art approaches perform poorly when applied to incomplete point clouds. This limitation of existing algorithms is particularly concerning since point clouds generated by 3D sensors in the real world are usually incomplete due to perspective view or occlusion by other objects. This paper proposes a general model for partial point clouds analysis wherein the latent feature encoding a complete point clouds is inferred by applying a local point set voting strategy. In particular, each local point set constructs a vote that corresponds to a distribution in the latent space, and the optimal latent feature is the one with the highest probability. This approach ensures that any subsequent point cloud analysis is robust to partial observation while simultaneously guaranteeing that the proposed model is able to output multiple possible results. This paper illustrates that this proposed method achieves state-of-the-art performance on shape classification, part segmentation and point cloud completion.

1 Introduction

The quality of 3D sensors has rapidly improved in recent years as their ability to accurately and quickly measure the depth of scenes has surpassed traditional vision-based methods [57, 13, 60, 22, 58]. This improved accessibility to point clouds demands the development of algorithms to interpret and analyze them. Inspired by the success of Deep Neural Networks (DNNs) in solving 2D image analysis tasks [16, 40, 26, 31, 19], approaches with DNNs have been successfully applied to perform similar point cloud analysis tasks such as shape classification and part segmentation [39, 29, 37, 30, 28, 52, 25]. These DNNs methods achieve state-of-the-art performance on these point cloud analysis tasks by learning representations from large synthetic datasets constructed from sampling the surfaces of CAD objects [53, 2]. Unfortunately since point clouds generated from 3D sensors in the real-world scenarios are often incomplete, these approaches struggle when tasked to perform shape classification or part segmentation on partial point clouds. Since real-world point clouds are often incomplete due to perspective of view or occlusions, this shortcoming of existing point cloud analysis methods is particularly cumbersome.

To address the limitations of existing approaches, this paper proposes a novel model for partial point clouds analysis. We model the point clouds as a partition of point sets. Each local point set independently contributes to infer the latent feature encoding the complete point cloud. In contrast to prior work on learning a representation of point clouds, we utilize an encoder to embed each local point set. All embeddings vote to infer a latent space characterized by a distribution. As

Preprint. Under review.

we show in this paper, giving each local point set a vote, ensures that the model has the ability to address the incomplete nature of real-world point clouds. Inspired by recent progress in variational inference [7, 24], we output a distribution for the latent variable and then use a decoder to generate a prediction from the latent value with the highest probability. In particular, each local point set generates a Gaussian distribution in the latent space and independently votes to form the distribution of the latent variable. This voting strategy ensures that the model outputs more accurate prediction when more partial observations are given, and the probabilistic modeling enables the model to generate multiple possible outputs.

The contributions of this paper are: (1) We propose that each local point set independently votes to infer the latent feature. This voting strategy is shown to be robust to partial observation; (2) We propose to construct each vote as a distribution in the latent space and this distribution modeling allows for diverse predictions; (3) The proposed model trained with complete point clouds performs robustly on partial observation at test, which reduces the cost of collecting large partial point clouds dataset; (4) The proposed model achieves state-of-the-art results on shape classification, part segmentation, and point clouds completion. In particular, it outperforms approaches trained with pairs of partial and complete point clouds on point clouds completion.

2 Related Work

To perform point cloud analysis, researchers have traditionally converted point clouds into 2D grids or 3D voxels since they can leverage existing convolutional neural networks (CNNs). With the help of CNNs, those approaches achieve impressive results on 3D shape analysis [10, 33, 38, 53]. Unfortunately, these 2D grid or 3D voxel representations degrade the resolution of objects. Researchers have attempted to address this issue by utilizing sparse representations [25, 48, 50]. However, these representations are still less efficient than point clouds and are unable to avoid quantization effects. More recently, PointNet [37] has pioneered the approaches of directly taking point clouds as inputs and processed them using DNNs. To accomplish this objective, it uses a symmetric function to aggregate information from each point which is transformed to a high-dimensional space. A variety of extensions have been applied to PointNet [39, 28, 30, 51, 59, 41]; however, none of them is robust to the partial observation that is common in real-world scenarios.

To address this challenge posed by partial observations, researchers have relied on training DNNs on partial point clouds collected in real-world scenarios [35, 54, 42, 36, 3, 20, 1, 5, 44, 12]. Each of these approaches rely on networks that were proposed to perform feature extraction on complete point clouds. Unfortunately, collecting and annotating those partial point datasets is expensive. Another approach seeks to first infer the missing data of the partially observed point clouds before later analysis. A variety of methods have been proposed to perform shape completion using distance fields and voxels [6, 15, 45, 27]. However, they degrade the resolution of objects represented by point clouds.

Recently, researches have been done to perform shape completion on point clouds. A common pipeline to perform this completion first encodes partial observations into a feature vector and then decodes it to complete point clouds. A variety of methods have been proposed for designing the decoder [9, 47, 55, 56, 49]. Multi-view methods [46, 17, 18] and approaches modeling an implicit distance function [34, 4] have also been applied to perform shape completion. However, each of these methods outputs a single prediction given inputs and lacks the ability to generate multiple plausible results. One notable exception is able to generate diverse results by modeling the spatial distribution of all the points [47]. Unfortunately, this approach is only able to address partial point clouds from specific locations. In contrast, the method developed in this paper has no such requirement on an observed partial point cloud, and leverages the distribution over the latent space to generate diverse predictions.

The variational autoencoder (VAE) is one of the popular methods to model a generative distribution [7]. It assumes a prior distribution of latent variables, which is often a Gaussian distribution. More recently, the conditional variational autoencoder (CVAE) was proposed to extend the VAE by modeling the conditional distribution [43]. Unfortunately, directly applying a CVAE to partial point clouds requires a collection of annotated partial point cloud datasets for training. This paper addresses this limitation by proposing each local point set serve as the unit voter to contribute to the latent feature. Encoding features learned for local point sets of complete point clouds can be leveraged for embedding local

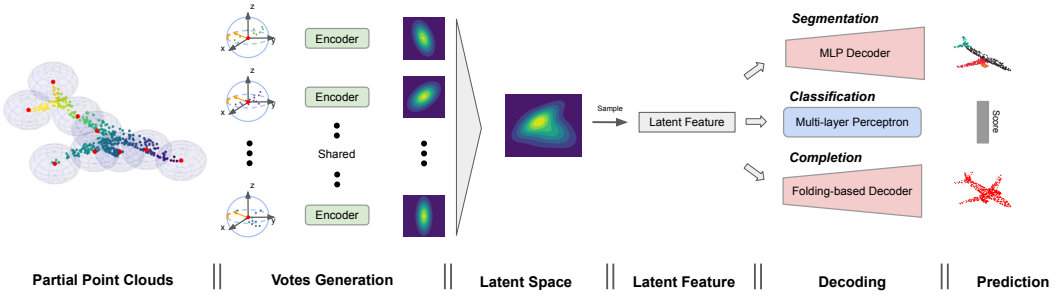


Figure 1: An illustration of the model developed in this paper. The input point clouds are modeled as a partition of point sets and each local point set is defined by its centroid and scale r. Local point sets are embedded by a shared-weight encoder. The voting via this encoder is used to infer the space for latent features. The latent feature with the highest probability is sampled from the inferred latent space and is then passed to a decoding module for prediction. We perform experiments on three tasks: classification, part segmentation and point clouds completion.

point sets of partial point clouds at test, which allows us to train on complete point clouds and perform on partial point clouds at test.

3 Problem Statement

Consider an observed partial point cloud, denoted by $\mathbf{x} \subseteq \mathbb{R}^3$. Suppose the output of a model for a point cloud analysis task is $\mathbf{y}$ and we are interested in modeling the conditional distribution $p(\mathbf{y}|\mathbf{x})$, which provides a way to predict $\mathbf{y}$ given the partial inputs $\mathbf{x}$. This paper aims to address the following three challenges: (1) classifying partial point clouds, (2) segmenting parts on partial point clouds, and (3) recovering complete point clouds from partial observation.

4 Method

This section describes the method we use to accomplish the aforementioned objective.

4.1 Preliminary: Conditional Variational Auto-encoder

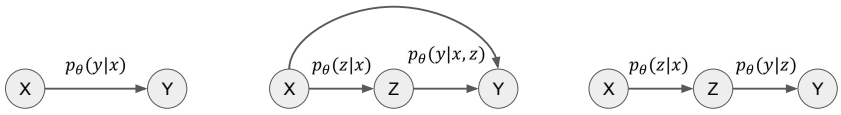
A CVAE [43] is a directed graphical model. The conditional generative process of CVAE is as follows: for a given observation $\mathbf{x}$, a noise vector $\mathbf{z}$ is drawn from the prior distribution $p_\theta(\mathbf{z}|\mathbf{x})$, and an output $\mathbf{y}$ is generated from the distribution $p_\theta(\mathbf{y}|\mathbf{x};\mathbf{z})$. The training objective, variational lower bound, of CVAE is written as follows:

$$\log p_\theta(\mathbf{y}|\mathbf{x}) \geq \mathcal{L}_{\phi,\theta}(\mathbf{y}|\mathbf{x}) = -\mathbf{KL}(q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})||p_\theta(\mathbf{z}|\mathbf{x}))) + E_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})}[\log p_\theta(\mathbf{y}|\mathbf{x},\mathbf{z})] \quad (1)$$

The CVAE is composed of recognition network $q_\phi(\mathbf{z}|\mathbf{x};\mathbf{y})$, prior network $p_\theta(\mathbf{z}|\mathbf{x})$, and generation network $p_\theta(\mathbf{y}|\mathbf{x};\mathbf{z})$. In this framework, the recognition network $q_\phi(\mathbf{z}|\mathbf{x};\mathbf{y})$ is used to approximate the prior network $p_\theta(\mathbf{z}|\mathbf{x})$. All distributions are modeled using neural networks. During training, the reparameterization trick [7] is applied to propagate the gradients of ϕ and θ through the latent variables $\mathbf{z}$.

4.2 Proposed Point Clouds Model

We model the point clouds $\mathbf{x}$ as an overlapping partition of point sets, denoted by $\{x_1, x_2, ..., x_N\}$ if there are N point sets in the partition. In the simplest setting, each point is described by just its 3D coordinates, i.e. $x_i \subseteq \mathbb{R}^3$. Each point set is defined by a centroid and scale, as is shown in the Figure 1. To evenly cover the whole point clouds, we use Farthest Point Sampling (FPS) [39] algorithm to sample centroids. The number of centroids and the scale are manually set.



(a) CNN (b) CVAE (c) Ours

Figure 2: An illustrative comparison between our proposed method and other conditional graphical models (CGMs).

4.3 Proposed Method

In contrast to CVAEs, in which the generation network takes $(\mathbf{x}, \mathbf{z})$ as inputs, we model the generation process as a Markov chain, as is shown in the Figure 2. Specifically, given the latent variable $\mathbf{z}$ sampled from $p(\mathbf{z}|\mathbf{x})$, $\mathbf{y}$ is independent on $\mathbf{x}$. As a result, the generation of the output $\mathbf{y}$ satisfies the following equation: $p(\mathbf{y}|\mathbf{x}, \mathbf{z}) = p(\mathbf{y}|\mathbf{z})$. The variational lower bound of this model is written as follows:

$$\mathcal{L}_{\phi,\theta}(\mathbf{y}|\mathbf{x}) = -\mathbf{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x}))) + E_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{y}|\mathbf{z})] \tag{2}$$

One problem with this learning framework is that the generation network $p_\theta(\mathbf{y}|\mathbf{z})$ takes values sampled from the recognition network $q_\phi(\mathbf{z}|\mathbf{x})$ at training while takes values sampled from the prior network $p_\theta(\mathbf{z}|\mathbf{x})$ at testing. This makes training inconsistent with testing. Similar to [43], we force consistency between these settings by making the recognition network $q_\phi(\mathbf{z}|\mathbf{x})$ the same as the prior network $p_\theta(\mathbf{z}|\mathbf{x})$. By doing this, $\mathbf{z}$ can be drawn from the distribution $q_\phi(\mathbf{z}|\mathbf{x})$ at both training and testing, and the KL divergence term becomes zero. We approximate the resulting version of Equation (2) with the Monte Carlo estimator formed by sampling $\mathbf{z}^{(i)}$ from the recognition network $q_\phi(\mathbf{z}|\mathbf{x})$:

$$\mathcal{L}_{\phi,\theta}(\mathbf{y}|\mathbf{x}) \approx \frac{1}{L}\sum_{i=1}^{L} \log p_\theta(\mathbf{y}|\mathbf{z}^{(i)}),\ \mathbf{z}^{(i)} \sim q_\phi(\mathbf{z}|\mathbf{x}). \tag{3}$$

Recall that in our case $\mathbf{x}$ is modeled as a set of local point sets$\{x_i\}_{i=1}^n$. Each of them generates a vote to compute the latent variable $\mathbf{z}$. This voting strategy is inspired by the Hough transform [8] and VoteNet [35], and it is shown to accumulate small bits of partial information and output confident predictions. Unlike VoteNet, which outputs each vote as a deterministic feature, we model each vote as a distribution in the latent space. By assuming the independence of each vote, the recognition network $q_\phi(\mathbf{z}|\mathbf{x})$ can be expanded as follows:

$$q_\phi(\mathbf{z}|\mathbf{x}) = \prod_{i=1}^{n} q_\phi(\mathbf{z}|x_i). \tag{4}$$

In the experiments, we assume $q_\phi(\mathbf{z}|x_i)$ is a Gaussian distribution characterized by mean vector and covariance matrix, which enables the use of a closed-form solution to optimizing $q_\phi(\mathbf{z}|\mathbf{x})$ with respect to $\mathbf{z}$. We denote the maximizing argument of $q_\phi(\mathbf{z}|\mathbf{x})$ by $\mathbf{z}_{opt}$. Equation (3) is equivalent to estimation with a single point when setting $L = 1$. Combining this with the highest probability sample of $\mathbf{z}$, given by $\mathbf{z}_{opt}$, the objective function can be further written as follows:

$$\mathcal{L}_{\phi,\theta}(\mathbf{y}|\mathbf{x}) \approx \log p_\theta(\mathbf{y}|\mathbf{z}_{opt}), \quad \text{where} \quad \mathbf{z}_{opt} = \underset{z}{\operatorname{argmax}} \prod_{i=1}^{n} q_\phi(\mathbf{z}|x_i). \tag{5}$$

Previous variational models choose latent features by sampling. However, $\mathbf{z}_{opt}$ in our case can be computed directly, so all operations are differentiated with respect to the parameters θ and ϕ, which means that the reparameterization trick is no longer needed.

Note that the loss function differs as the task changes. A common softmax cross-entropy loss is used for classification and segmentation whereas the Chamfer distance [9] is used for training partial point clouds completion. After training, the generative process is as follows: for the given observation $\mathbf{x}$,

Method	Input	Complete	Partial
PointNet [37]	xyz	88.8	20.9
PointNet++ [39]	xyz	91.0	61.5
RS-CNN [30]	xyz	92.3	43.3
DG-CNN [51]	xyz	**92.9**	51.5
Ours	xyz	91.4	**86.4**

Table 1: Classification accuracy on ModelNet40 dataset. Overall classification accuracy is reported on complete point clouds (complete) and simulated partial point clouds (partial).

$\mathbf{z}_{opt}$ is the result of voting from all $q_\phi(\mathbf{z}|x_i)$, and then the output $\mathbf{y}$ is generated from $p_\theta(\mathbf{y}|\mathbf{z}_{opt})$. The use of $\mathbf{z}_{opt}$ produces a single deterministic prediction. Diverse predictions are generated by instead sampling $\mathbf{z}^{(i)} \sim q_\phi(\mathbf{z}|\mathbf{x})$ followed by applying the generation network as in the deterministic case.

5 Experiment

5.1 Implementation

Network architecture The architecture of the proposed model is illustrated in the Figure 1. We use DNNs to model $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{y}|\mathbf{z})$. A shared-weights network is used to represent $q_\phi(\mathbf{z}|x_i)$. Given the local point set, we represent each point within the local region relative to the centroid and then use a shared-weight PointNet as the basic feature extractor to encode the local region. Both the encoding feature and coordinates of centroid are processed by a multi-layer perceptron (MLP) and it outputs a distribution of the latent space. We assume a simple case of multivariate Gaussian distribution with diagonal covariance matrix. Different analysis tasks correspond to different networks for modeling $p_\theta(\mathbf{y}|\mathbf{z})$ as is shown in Figure 1. A folding-based decoder [55] is used for point clouds completion.

Training During training, we partition the point cloud into 64 local point sets, and each of them generates a vote in the latent space. To make the voting strategy tenable, we propose that a random number of votes are selected to contribute to latent feature at training and in the extreme case only one vote is selected. This ensures that a single vote has the potential to be decoded to the prediction and the trained model is robust to any type partialness. All votes contribute to compute latent feature at testing.

Simulated partial Point clouds For datasets that do not provide partial point clouds, we simulate partial point clouds during testing. Partial point clouds are synthesized by selecting points falling into one side of a plane. The plane goes through the origin and is defined by the normal, which is a 3D vector and generated by randomly sampling from a normal distribution. Note that these simulated partial point clouds are only used during testing. Sample partial point clouds are visualized in the Figure 6.

5.2 Point Clouds Classification

We first consider the task of point cloud classification. This requires that the model extract global features describing distinct geometric information and decode it into a predicated category.

Dataset We use the ModelNet40 [53] dataset to evaluate our proposed method on shape classification of point clouds. It contains 12,311 CAD models from 40 object categories. In the experiments, point clouds are generated by evenly sampling 1024 points from the surface of objects. We follow the same strategy of set splitting as in [37]. Before being passed into the model, point clouds are centered and normalized within a unit sphere. No data augmentation techniques are applied during training.

Results Quantitative results on Modelnet40 are shown in the Table 1. Overall classification accuracy is reported on both complete point clouds and simulated partial point clouds. All listed methods achieve the state-of-the-art results on complete point clouds and our proposed method performs slightly better than PointNet and PointNet++. When evaluated on simulated partial point clouds, however, other approaches experience a significant drop in accuracy. This is unsurprising since existing approaches are designed for complete point clouds, and generalize poorly to novel partial point clouds. In contrast, our proposed method trained on complete point clouds is robust to partial observation and achieves 86.4% classification accuracy on partial point clouds.

Figure 3: Qualitative results on part segmentation of point clouds using the method presented in this paper. Different colors correspond to distinct segments. Left: segmentation on the ShapeNet part dataset. Middle: segmentation on the simulated partial point clouds in the ShapeNet. Right: segmentation on the Completion3D dataset.

Method	Input	Complete	Partial
PointNet [37]	xyz	80.5	29.9
PointNet++ [39]	xyz	82.0	30.9
DG-CNN [51]	xyz	82.3	29.8
RS-CNN [30]	xyz	**82.4**	30.6
Ours	xyz	79.0	**78.1**

Table 2: Part segmentation results on ShapeNet part dataset. Mean intersection of unions (mIoUs) is reported on complete point clouds (Complete) and simulated partial point clouds (Partial).

5.3 Part Segmentation

Given the point clouds and object category, the part segmentation tasks requires predicting a part label for each point. As a result, this task requires that the model extract both global and local information.

Dataset We use the ShapeNet part dataset [53] to evaluate our proposed method on part segmentation of point clouds. It contains 16,881 shapes from 16 object categories with 50 parts. Each point cloud contains 2048 points which are generated by evenly sampling from the surface of objects. We follow the same set splitting conventions in [39]. During evaluation, mean inter-over-union (mIoU) that are averaged across all classes is reported. No data augmentation techniques are applied during training.

Results We report the results of our proposed method and compare its performance to existing approaches in the Table 2. Our proposed method has superior performance on partial point clouds and achieves 78.1 mIoU, while others achieve around 30 mIoU. This robustness to partial observation comes at the expense of slightly lower accuracy on segmentation of complete point clouds when compared with other approaches. We also test the robustness of our approach by applying it to the completion3D dataset [49] which contains partial point clouds but no part labels. Qualitative results on part segmentation by applying the method proposed in this paper are shown in the Figure 3.

5.4 Point Clouds Completion

Given partial point clouds, point cloud completion tries to generate points to fill in the missing parts of the input. This requires a model that has the ability to infer a global feature which encodes complete point clouds from the partial inputs.

Dataset We evaluate our model on Completion3D [49], which is a 3D object point cloud completion benchmark. It contains pairs of partial and complete point clouds from 8 categories which are derived from the Shapenet dataset with 2048 points per object point clouds. We apply the set splitting given by the dataset. Partial point clouds are generated by back-projecting 2.5D depth images into 3D space and complete point clouds are used as ground truth.

Results Quantitative results on Completion3D's withheld test dataset are shown in Table 3 where the Chamfer distance multiplied by 10^4 is reported. Our proposed model outperforms FoldingNet

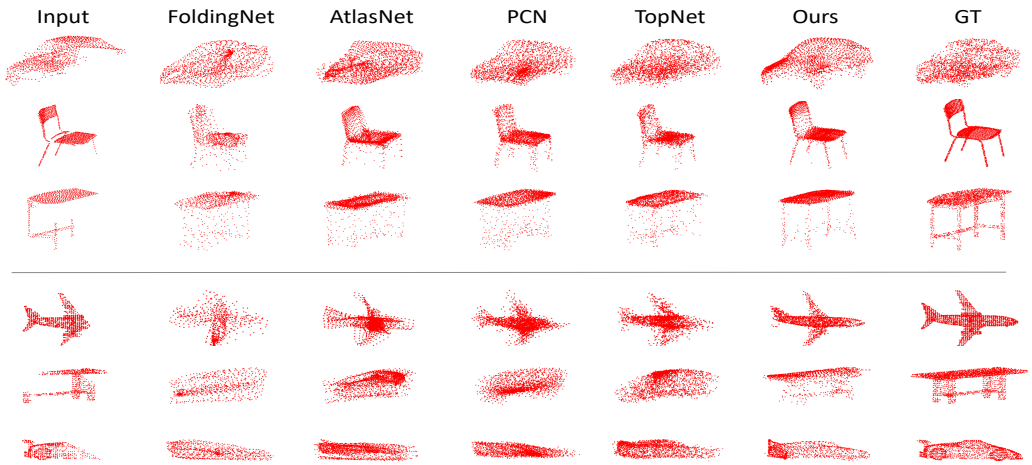


Figure 4: Qualitative point clouds completion results on Completion3D. Top: completion results on partial point clouds in Completion3D. Bottom: completion results on simulated partial point clouds introduced in the paper.

Figure 5: Completion on real-world point clouds. Top: input partial point clouds. Bottom: complete point clouds generated by ours. Partial point clouds are generated from ScanNet.

Model	Average CD
FoldingNet [55]	19.07
PCN [56]	18.22
AtlasNet [14]	17.77
TopNet [49]	**14.25**
Ours	18.18

Table 3: The performance of various state-of-the-art algorithms on partial point cloud completion on the Completion3D benchmark dataset. Results are reported on the Completion3D's withheld test set. The average Chamfer distance (CD) over classes in the test set is reported, multiplied by 10^4.

Model	Average CD
FoldingNet [55]	34.56
PCN [56]	34.93
AtlasNet [14]	39.73
TopNet [49]	31.87
Ours	**17.22**

Table 4: The performance of various state-of-the-art algorithms trained via the Completion3D dataset on partial point cloud completion on a simulated partial point cloud dataset. Results are reported on simulated partial point clouds of validation set. Average Chamfer distance (CD) over classes is reported, multiplied by 10^4.

and PCN, which are trained with both partial and complete point clouds, while only complete point clouds are used during training for the method developed in this paper.

Since Completion3D's withheld test dataset has only a limited number of partial cloud examples, we more exhaustively evaluated all approaches by experimenting on simulated partial point clouds as in Figure 3 using models that were each trained on the Completion3D. As shown in Table 4, all approaches except the one developed in this paper experience a significant performance drop. We suspect that this is because the partial point clouds in the Completion3D are different from the simulated partial point clouds. However, our proposed method generalizes well to any partial point cloud. Qualitative completion results are shown in Figure 4, and it shows that we have sharper output point clouds. More challenging tests are performed on real-world point clouds and are shown in the Figure 5. The partial point clouds are extracted within the object bounding boxes provided in ScanNet [5] and each point cloud is transformed to the box's coordinates.

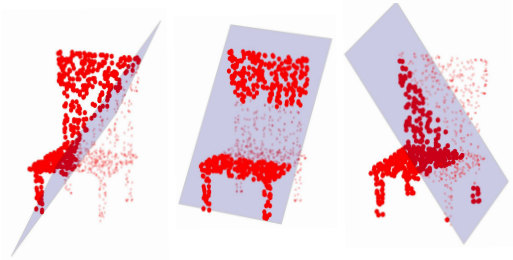


Figure 6: Sample simulated partial point clouds. A random 2D plane is generated to select which points (bolded red) form the partial point cloud.

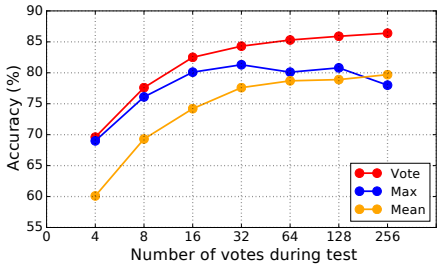


Figure 7: Results of different aggregation strategies for computing latent features on simulated partial point clouds in ModelNet40.

5.5 Voting Strategy Analysis

The process of computing latent feature is inspired by the voting mechanism proposed in [35]. As proposed in that paper, we infer the latent space by votes from local point sets. However, the vote in our case is a distribution in the latent space instead of a deterministic feature vector. The optimal latent feature is the sampled one with the highest probability. In this section, we analyze this voting strategy and compare it with an aggregation strategy where the extracted features from local point sets are aggregated using a symmetric function. We study two different symmetric functions: max pooling and mean pooling. To utilize the aggregation strategy, we change the vote of each local point set from a distribution to a deterministic feature vector.

The comparison results are shown in the Figure 7. The evaluation metric is the average classification accuracy on the simulated partial ModelNet40. All models are trained using the proposed training strategy where a random number of votes less than or equal to 10 are selected to contribute to latent features. Our proposed voting strategy is verified by the improved accuracy when compared to the aggregation strategy using either max pooling or mean pooling. Classification accuracy grows as the number of votes increases during testing in both the voting strategy and mean pooling. This is because more votes accumulate more information for prediction. However, this is not the case for max pooling, whose accuracy peaks at 32 votes. This indicates that max pooling is sensitive to the number of selected votes.

5.6 Visualization of multiple predictions

The proposed model is designed to be able to generate multiple possible outputs. This is achieved by the latent space model. The latent space is represented by a set of multivariate Gaussian distributions generated by local point sets. As a result, the most possible prediction is decoded from the latent value with the highest probability. However, diverse prediction can be generated by sampling from the latent space. We visualize the results in Figure 8.

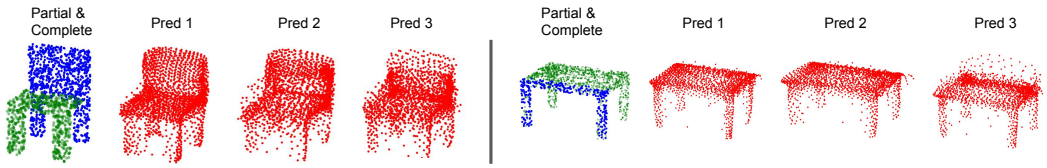


Figure 8: Visualization of diverse predictions on point clouds completion. Blue points are input partial point clouds, green points are missing and red points are predictions from our proposed method.

6 Conclusion

This paper proposes a general model for partial point clouds analysis. In particular, point clouds are modeled as a partition of point sets which generate votes to model a latent space distribution. This voting strategy is shown to accumulate partial information and be robust to partial observation. The

sampled latent feature in the latent space is then decoded for prediction. Extensive experiments are performed in classification, part segmentation, and completion and state-of-the-art results on all of them demonstrate the effectiveness of the proposed method.

7 Acknowledgment

This work was supported by a grant from Ford Motor Company via the Ford-UM Alliance under award N028603.

References

[1] Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S.: 3d semantic parsing of large-scale indoor spaces. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1534–1543 (2016)

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

[3] Chen, Y., Liu, S., Shen, X., Jia, J.: Fast point r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9775–9784 (2019)

[4] Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5939–5948 (2019)

[5] Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE (2017)

[6] Dai, A., Ruizhongtai Qi, C., Nießner, M.: Shape completion using 3d-encoder-predictor cnns and shape synthesis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5868–5877 (2017)

[7] Diederik, P.K., Welling, M.: Auto-encoding variational bayes. In: Proceedings of the International Conference on Learning Representations (ICLR). vol. 1 (2014)

[8] Duda, R.O., Hart, P.E.: Use of the hough transformation to detect lines and curves in pictures. Communications of the ACM **15**(1), 11–15 (1972)

[9] Fan, H., Su, H., Guibas, L.J.: 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. pp. 605–613 (2017)

[10] Feng, Y., Zhang, Z., Zhao, X., Ji, R., Gao, Y.: Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 264–272 (2018)

[11] Fey, M., Lenssen, J.E.: Fast graph representation learning with PyTorch Geometric. In: ICLR Workshop on Representation Learning on Graphs and Manifolds (2019)

[12] Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3354–3361. IEEE (2012)

[13] Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 270–279 (2017)

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

[15] Han, X., Li, Z., Huang, H., Kalogerakis, E., Yu, Y.: High-resolution shape completion using deep neural networks for global structure and local geometry inference. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 85–93 (2017)

[16] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

[17] Hu, T., Han, Z., Shrivastava, A., Zwicker, M.: Render4completion: Synthesizing multi-view depth maps for 3d shape completion. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 0–0 (2019)

[18] Hu, T., Han, Z., Zwicker, M.: 3d shape completion with multi-view consistent inference. arXiv preprint arXiv:1911.12465 (2019)

[19] Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708 (2017)

[20] Huang, Q., Wang, W., Neumann, U.: Recurrent slice networks for 3d segmentation of point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2626–2635 (2018)

[21] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning. pp. 448–456 (2015)

[22] Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., Bry, A.: End-to-end learning of geometry and context for deep stereo regression. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 66–75 (2017)

[23] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[24] Kingma, D.P., Welling, M.: An introduction to variational autoencoders. arXiv preprint arXiv:1906.02691 (2019)

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

[26] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)

[27] Le, T., Duan, Y.: Pointgrid: A deep network for 3d shape understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 9204–9214 (2018)

[28] Li, J., Chen, B.M., Hee Lee, G.: So-net: Self-organizing network for point cloud analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 9397–9406 (2018)

[29] Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: Pointcnn: Convolution on x-transformed points. In: Advances in Neural Information Processing Systems. pp. 820–830 (2018)

[30] Liu, Y., Fan, B., Xiang, S., Pan, C.: Relation-shape convolutional neural network for point cloud analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8895–8904 (2019)

[31] Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3431–3440 (2015)

[32] Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: Proc. icml. vol. 30, p. 3 (2013)

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

[34] Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 165–174 (2019)

[35] Qi, C.R., Litany, O., He, K., Guibas, L.J.: Deep hough voting for 3d object detection in point clouds. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9277–9286 (2019)

[36] Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum pointnets for 3d object detection from rgb-d data. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 918–927 (2018)

[37] Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 652–660 (2017)

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

[39] Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in neural information processing systems. pp. 5099–5108 (2017)

[40] Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. pp. 91–99 (2015)

[41] Shen, Y., Feng, C., Yang, Y., Tian, D.: Mining point cloud local structures by kernel correlation and graph pooling. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4548–4557 (2018)

[42] Shi, S., Wang, X., Li, H.: Pointrcnn: 3d object proposal generation and detection from point cloud. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–779 (2019)

[43] Sohn, K., Lee, H., Yan, X.: Learning structured output representation using deep conditional generative models. In: Advances in neural information processing systems. pp. 3483–3491 (2015)

[44] Song, S., Lichtenberg, S.P., Xiao, J.: Sun rgb-d: A rgb-d scene understanding benchmark suite. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 567–576 (2015)

[45] Stutz, D., Geiger, A.: Learning 3d shape completion from laser scan data with weak supervision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1955–1964 (2018)

[46] Su, H., Maji, S., Kalogerakis, E., Learned-Miller, E.: Multi-view convolutional neural networks for 3d shape recognition. In: Proceedings of the IEEE international conference on computer vision. pp. 945–953 (2015)

[47] Sun, Y., Wang, Y., Liu, Z., Siegel, J., Sarma, S.: Pointgrow: Autoregressively learned point cloud generation with self-attention. In: The IEEE Winter Conference on Applications of Computer Vision. pp. 61–70 (2020)

[48] Tatarchenko, M., Dosovitskiy, A., Brox, T.: Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2088–2096 (2017)

[49] Tchapmi, L.P., Kosaraju, V., Rezatofighi, H., Reid, I., Savarese, S.: Topnet: Structural point cloud decoder. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 383–392 (2019)

[50] Wang, P.S., Liu, Y., Guo, Y.X., Sun, C.Y., Tong, X.: O-cnn: Octree-based convolutional neural networks for 3d shape analysis. ACM Transactions on Graphics (TOG) **36**(4), 72 (2017)

[51] Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics (TOG) **38**(5), 146 (2019)

[52] Wu, W., Qi, Z., Fuxin, L.: Pointconv: Deep convolutional networks on 3d point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9621–9630 (2019)

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

[54] Yang, B., Wang, J., Clark, R., Hu, Q., Wang, S., Markham, A., Trigoni, N.: Learning object bounding boxes for 3d instance segmentation on point clouds. In: Advances in Neural Information Processing Systems. pp. 6737–6746 (2019)

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

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

[57] Žbontar, J., LeCun, Y.: Stereo matching by training a convolutional neural network to compare image patches. The journal of machine learning research **17**(1), 2287–2318 (2016)

[58] Zhang, J., Skinner, K.A., Vasudevan, R., Johnson-Roberson, M.: Dispsegnet: Leveraging semantics for end-to-end learning of disparity estimation from stereo imagery. IEEE Robotics and Automation Letters **4**(2), 1162–1169 (2019)

[59] Zhang, Y., Rabbat, M.: A graph-cnn for 3d point cloud classification. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 6279–6283. IEEE (2018)

[60] Zhou, T., Brown, M., Snavely, N., Lowe, D.G.: Unsupervised learning of depth and ego-motion from video. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1851–1858 (2017)

Supplementary

8 Computation of the Optimal Latent Feature

Here we provide a derivation of $\mathbf{z}_{opt}$, the optimal latent feature with highest probability in the latent space. The distribution of the latent space $q_\phi(\mathbf{z}|\mathbf{x})$ is represented by a set of multivariate Gaussian distributions (Equation (5) in the main paper). By assuming that votes are independent and $q_\phi(\mathbf{z}|x_i)$ is Gaussian distributed, the derivation of $\mathbf{z}_{opt}$ is as follows:

$$\begin{aligned}
\mathbf{z}_{opt} &= \underset{z}{\operatorname{argmax}}\ q_\phi(\mathbf{z}|\mathbf{x}) \\
&= \underset{z}{\operatorname{argmax}}\ \log(q_\phi(\mathbf{z}|\mathbf{x})) \\
&= \underset{z}{\operatorname{argmax}}\ \sum_{i=1}^{n} \log(q_\phi(\mathbf{z}|x_i)) \\
&= \underset{z}{\operatorname{argmax}}\ \sum_{i=1}^{n} \log\left(\frac{1}{(2\pi)^{m/2}|\Sigma|_i^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{z}-\mu_i)^T\Sigma_i^{-1}(\mathbf{z}-\mu_i)\right)\right) \\
&= \underset{z}{\operatorname{argmax}}\ -\sum_{i=1}^{n} \log\left((2\pi)^{m/2}|\Sigma|_i^{1/2}\right) + \sum_{i=1}^{n}\left(-\frac{1}{2}(\mathbf{z}-\mu_i)^T\Sigma_i^{-1}(\mathbf{z}-\mu_i)\right) \\
&= \underset{z}{\operatorname{argmax}}\ \sum_{i=1}^{n}\left(-\frac{1}{2}(\mathbf{z}-\mu_i)^T\Sigma_i^{-1}(\mathbf{z}-\mu_i)\right)
\end{aligned} \tag{6}$$

where a multivariate Gaussian distribution is characterized by mean vector μ_i and covariance matrix Σ_i; n is the number of votes; m is the dimension of the latent space. The solution to optimizing $q_\phi(\mathbf{z}|\mathbf{x})$ can be computed by setting derivative to zero:

$$\begin{aligned}
0 &= \frac{\partial \sum_{i=1}^{n}\left(-\frac{1}{2}(\mathbf{z}-\mu_i)^T\Sigma_i^{-1}(\mathbf{z}-\mu_i)\right)}{\partial \mathbf{z}} \\
&= \sum_{i=1}^{n}\Sigma_i^{-1}(\mathbf{z}-\mu_i)
\end{aligned} \tag{7}$$

Thanks to concavity, the maximizing argument $\mathbf{z}_{opt}$ of $q_\phi(\mathbf{z}|\mathbf{x})$ is given by:

$$\mathbf{z}_{opt} = \frac{\sum_{i=1}^{n}\Sigma_i^{-1}\mu_i}{\sum_{i=1}^{n}\Sigma_i^{-1}} \tag{8}$$

For simplicity, we assume diagonal covariance matrix during experiments. Both μ_i and Σ_i are generated from each local point set, and modeled by neural networks.

9 Details of Implementation

9.1 Training

We implement our network in PyTorch and use PyTorch Geometric Library [11]. During optimization, we use the Adam optimizer [23] with default parameters except for the learning rate. We train models for three different tasks. (1) For point clouds classification experiments, the learning rate starts with 0.001 and is scaled by 0.2 every 200 epochs and total 500 epochs are performed. Batch size is 64 and we split them into 2 NVIDIA Tesla V100 GPUs during training. (2) For part segmentation experiments, the learning rate starts with 0.001 and is scaled by 0.2 every 200 epochs and total 500 epochs are performed. Batch size is 128 and we split them into 4 NVIDIA Tesla V100 GPUs during training. (3) For point clouds completion experiments, the learning rate starts with 0.0002 and is scaled by 0.2 every 200 epoch and total 500 epochs are performed. Batch size is 64 and we split them into 4 NVIDIA Tesla V100 GPUs during training.

Model	Plane	Cabinet	Car	Chair	Lamp	Sofa	Table	W.craft	Average
FoldingNet [55]	12.83	23.01	14.88	25.69	21.79	21.31	20.71	11.51	19.07
PCN [56]	9.79	22.70	12.43	25.14	22.72	20.26	20.27	11.73	18.22
AtlasNet [14]	10.36	23.40	13.40	24.16	20.24	20.82	17.52	11.62	17.77
TopNet [49]	7.32	**18.77**	**12.88**	**19.82**	**14.60**	**16.29**	**14.89**	**8.82**	**14.25**
Ours	**6.88**	21.18	15.78	22.54	18.78	28.39	19.96	11.16	18.18

Table 5: The performance of various state-of-the-art algorithms on partial point cloud completion on the Completion3D benchmark dataset. Results are reported on the Completion3D's withheld test set. The Chamfer distance (CD) is reported, multiplied by 10^4.

Model	Plane	Cabinet	Car	Chair	Lamp	Sofa	Table	W.craft	Average
FoldingNet [55]	25.79	40.52	16.12	39.90	43.01	43.76	40.88	26.54	34.56
PCN [56]	21.58	41.87	16.56	38.86	50.19	43.37	39.44	27.57	34.93
AtlasNet [14]	23.13	49.60	16.80	43.34	60.83	48.21	41.94	33.96	39.73
TopNet [49]	21.61	**15.24**	38.14	35.23	44.42	38.36	36.18	25.97	31.87
Ours	**7.54**	17.96	**9.22**	**19.49**	**29.97**	**15.82**	**24.58**	**13.15**	**17.22**

Table 6: The performance of various state-of-the-art algorithms trained via the Completion3D dataset on partial point cloud completion on a simulated partial point cloud dataset. Results are reported on simulated partial point clouds of validation set. The Chamfer distance (CD) is reported, multiplied by 10^4.

9.2 Network Architecture

We use similar notations as PointNet++ [39] to describe the network architecture of the proposed model. $SA(k, r, [l_0, l_1..., l_d])$ is a set abstraction (SA) level with k local regions of ball radius r using a shared-weights PointNet structure [37], which contains d fully connected layers $l_i (i = 1, ..., d)$ and l_0 is the width of inputs. $FC([l_0, l_1], dp)$ represents a fully connected layer with input width l_0, output width l_1, and dropout ratio dp. All layers are followed by batch normalization [21] and Leaky ReLU [32] layers except for the last prediction layer, last layer in the vote generation, and layers within the folding-based decoder.

Points coordinates are first transformed to a high-dimensional space by a fully connected layer. For all experiments, the architecture in vote generation process is the same and the outputs are the stack of mean vectors and diagonal elements of covariance matrix, since we model the vote as a multivariate Gaussian distribution:

$$FC([3, 64]) \longrightarrow SA(64, 0.2, [64 + 3, 64, 128, 512]) \longrightarrow FC([512 + 3, 512, 1024 * 2])$$

For shape classification experiments, the architecture for decoding the latent feature into K category scores is as follows:

$$FC([1024, 512], 0.5) \longrightarrow FC([512, 256], 0.5) \longrightarrow FC([256, K])$$

For part segmentation experiments, the encoding feature for each point is the stack of the latent feature, transformed point coordinates, and a one-hot vector for representing the object category. The architecture for point-wise prediction of K part category scores is as follows:

$$FC([1024{+}64{+}16, 512], 0.5) \longrightarrow FC([512, 256], 0.5) \longrightarrow FC([256, 128], 0.5) \longrightarrow FC([128, K])$$

For point clouds completion experiments, model with 0.1 ball radius achieves the best performance. The architecture of decoder is inspired by the folding idea proposed in [55], which folds 2D grids into 3D shapes:

$$FC([1024 + 2, 512, 512, 3]) \longrightarrow FC([1024 + 3, 512, 512, 3])$$

10 Point Clouds Completion

The quantitative results on partial point clouds completion on Completion3D test set is shown in the Table 5. With only training on complete point clouds, our proposed model still outperforms FoldingNet and PCN, which are trained with both partial and complete point clouds.

Model	BN	DP	# v. train	# v. test	radius	Bottleneck	Acc.
A_1			10	16	0.25	1024	78.4
A_2	✓		10	16	0.25	1024	80.9
A_3	✓	✓	10	16	0.25	1024	**81.0**
B_1	✓	✓	4	16	0.25	1024	79.1
B_2	✓	✓	10	16	0.25	1024	**81.0**
B_3	✓	✓	16	16	0.25	1024	79.4
B_4	✓	✓	32	16	0.25	1024	78.1
B_5	✓	✓	64	16	0.25	1024	76.2
C_1	✓	✓	10	8	0.25	1024	76.8
C_2	✓	✓	10	16	0.25	1024	81.0
C_3	✓	✓	10	32	0.25	1024	82.7
C_4	✓	✓	10	64	0.25	1024	83.8
C_5	✓	✓	10	128	0.25	1024	84.4
C_6	✓	✓	10	256	0.25	1024	**84.6**
D_1	✓	✓	10	256	0.15	1024	83.1
D_2	✓	✓	10	256	0.20	1024	**86.4**
D_3	✓	✓	10	256	0.25	1024	84.6
D_4	✓	✓	10	256	0.35	1024	82.5
E_1	✓	✓	10	256	0.20	512	85.3
E_2	✓	✓	10	256	0.20	1024	86.4
E_3	✓	✓	10	256	0.20	2048	**86.8**

Table 7: Ablation study. The metric is overall classification accuracy on the simulated partial ModelNet40 test set. “BN" indicates using batch normalization; “DP" indicates using the dropout technique in fully connected layers except the final one; “# v. train" indicates the maximum number of votes selected to contribute to the latent feature at training; “# v. test" indicates the number of votes at test; “radius” indicates the ball radius of local regions; “Bottleneck" indicates the dimension of the latent space.

We further evaluate all approaches on simulated partial point clouds using models trained on the Completion3D dataset. The simulated partial point clouds are constructed by processing the validation set and selecting points falling into one side of a random plane. As it is shown in the Table 6, all approaches except the one developed in this paper experience a significant performance drop. We suspect that this is due to the difference between partial point clouds in Completion3D and the simulated partial point clouds. However, the proposed method achieves similar performance on both partial point clouds and this leads to the conclusion that the proposed method has a better generalizability to any partial point clouds.

11 Ablation Study

Results of ablation study are shown in the Table 7 and the overall classification accuracy is reported on the simulated partial test set of ModelNet40. Unsurprisingly, the accuracy of models is improved after using the batchnorm and dropout techniques, from 78.4% to 81.0%, shown by Model A_i. We model the point clouds as an overlapping partition of point sets, each of which is defined by its centroid and scale. As it is shown by Model D_i, the performance of this modeling is sensitive to the scale and accuracy peaks at 0.2 (86.4%). This can be in part explained by that local regions with small scale contain inefficient distinct geometry features to infer the latent feature encoding the complete point clouds, while the learned features for local regions with large scale tend to be different from those in partial point clouds with unexpected edges due to missing parts. Model E_i illustrates that the performance of model grows as we increase the dimension of the latent space (bottleneck), and it saturates at 2048 (86.8%).

We propose that the latent feature encoding the complete point clouds are inferred by voting from local point sets. To make this voting strategy tenable, we design a training strategy that random number of votes are selected to compute the latent feature. Specifically, the maximum number of selected votes is manually set during training. As it is shown by Model B_i, the performance peaks when the maximum number of votes is set to 10 (81.0%). Model C_i illustrates that the accuracy

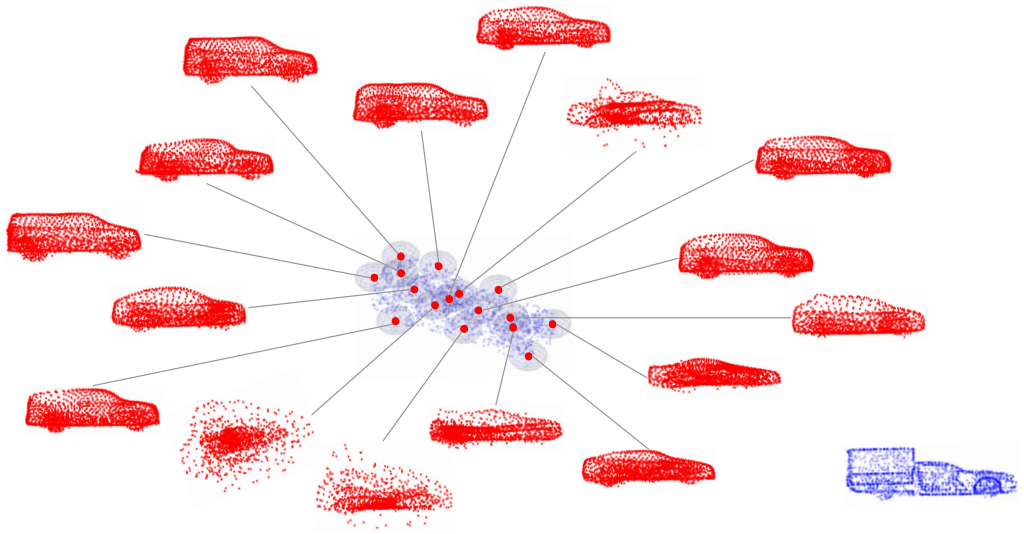


Figure 9: Visualization of voting strategy. The optimal latent feature inferred by a single vote is decoded into complete point clouds using a decoding module as in the voting case. The blue point clouds shows the complete point clouds.

of the trained model grows as the number of votes increases at test from 76.8% to 84.6%, which indicates that more votes accumulate more information for prediction.

12 Visualization of Voting Strategy

The latent space proposed in this paper is represented by a set of independent multivariate Gaussian distributions generated from local point sets. Combining this with the designed training strategy, each local point set is able to infer a distribution in the latent space. We perform experiments on partial point clouds completion on the Completion3D dataset and visualize each vote. Specifically, the optimal latent feature inferred by a single vote is decoded into complete point clouds using the folding-based decoder as in the voting case. As it is shown in the Figure 9, local point sets located at different parts of object generate votes encoding complete point clouds with different shapes. In the shown example, votes at the front of the vehicle tend to infer vehicles with sloping rears, while votes at the rear tend to infer truck-like vehicles. Moreover, compared to votes located at the front and the rear of the vehicle, votes in the middle contain less distinct geometry information since their decoded point clouds are blurrier.

13 Visualization of Multiple Predictions

The method developed in this paper is designed to be able to generate multiple possible outputs, and this is achieved by modeling the latent space. The latent space is represented by a set of multivariate Gaussian distributions generated by local point sets. As a result, the most possible prediction is decoded from the latent value with the highest probability. However, diverse predictions can be generated by sampling from the latent space and followed by the decoding module. Since it is not easy to sample in the latent space as it is represented by a set of distributions, we instead sample latent value by interpolating between the optimal latent feature inferred by all votes and the optimal latent feature inferred by a single vote. We perform experiments on point clouds completion and visualize the results in Figure 11.

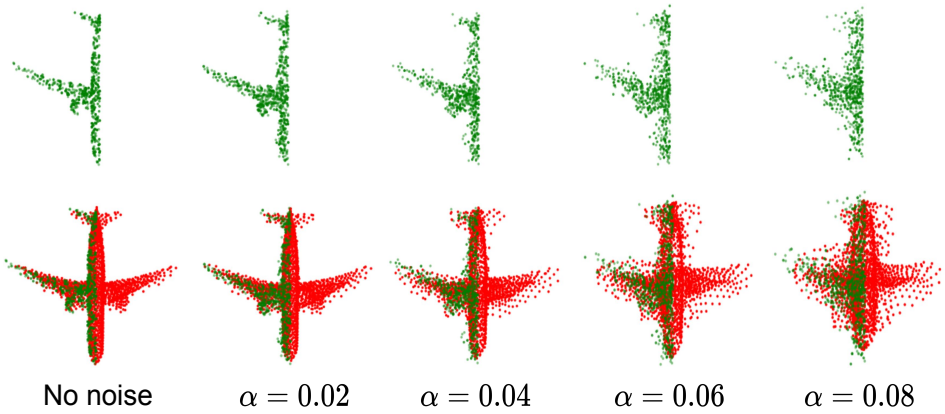


Figure 10: Results of point clouds completion obtained from the noisy partial observation. Gaussian noise with zero mean is assumed and the standard deviation is indicated at the bottom. Top: input partial observation. Bottom: prediction (red) overlapped with inputs (green).

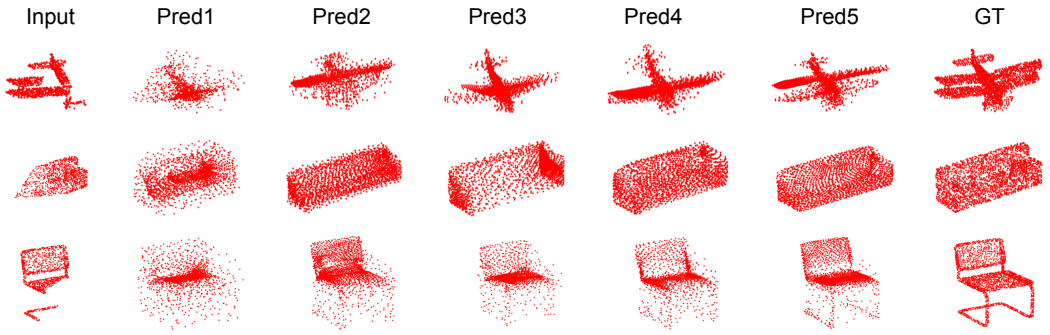


Figure 11: Visualization of diverse predictions on point clouds completion on the Completion3D dataset.

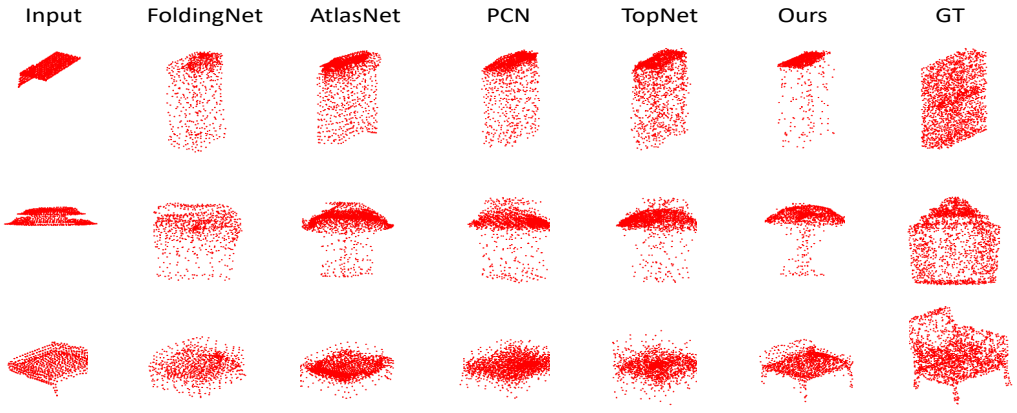


Figure 12: Failure cases of point clouds completion on the Completion3D dataset.

14 Visualization of Point Clouds Completion with Noisy Inputs

We visualize the results of point clouds completion with added noise in the Figure 10. Input partial point clouds are perturbed using Gaussian noise with zero mean, and the standard deviation differs in experiments as they are indicated at bottom of the figure.

15 Failure Cases on Point Clouds Completion

We show failure cases of point clouds completion on Completion3D in the Figure 12. Given partial observation with no distinct geometric information, all models fail to generate correct complete point clouds. However, the method developed in this paper is able to generate sharp and reasonable completion, while outputs of other approaches are blurry. We suspect the reason is that other approaches will generate mean shape of what they have trained on when difficult partial point clouds are observed.