Supplementary: Point Set Voting for Partial Point Cloud Analysis

Junming Zhang[1], Weijia Chen[2], Yuping Wang[1], Ram Vasudevan[3] and Matthew Johnson-Roberson[4]

I. 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:

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

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:

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

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

$$\mathbf{z}_{opt} = \frac{\sum_{i=1}^{n}\Sigma_i^{-1}\mu_i}{\sum_{i=1}^{n}\Sigma_i^{-1}} \tag{3}$$

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.

II. Details of Implementation

A. Training

We implement our network in PyTorch and use PyTorch Geometric Library [1]. During optimization, we use the Adam optimizer [2] 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.

[1]J. Zhang and Y. Wang are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA junming, ypw@umich.edu
[2]W. Chen is with the Robotics Program, University of Michigan, Ann Arbor, MI 48109 USA weijiac@umich.edu
[3]R. Vasudevan is with the Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI 48109 USA ramv@umich.edu
[4]M. Johnson-Roberson is with the Department of Naval Architecture and Marine Engineering, University of Michigan, Ann Arbor, MI 48109 USA mattjr@umich.edu

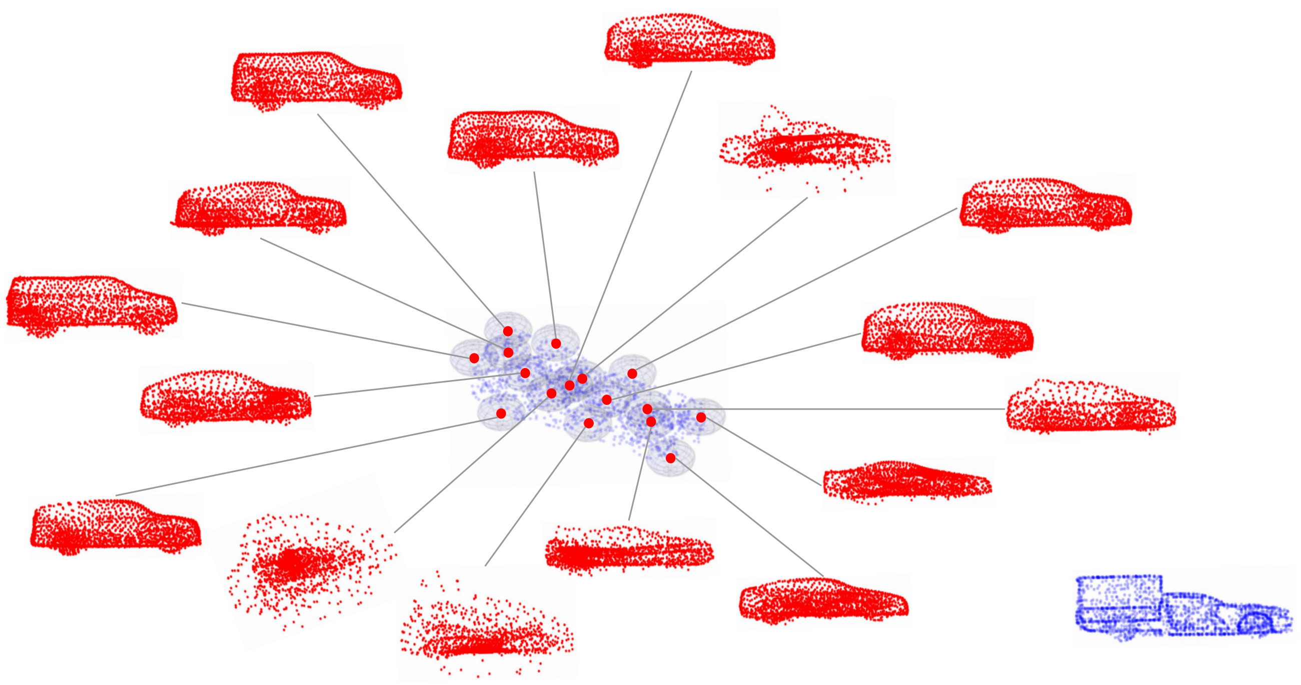

Fig. 1: Visualization of voting strategy. The optimal latent feature inferred by a single vote is decoded into complete point clouds using a decoding module as in the voting case. The blue point clouds shows the complete point clouds.

B. *Network Architecture*

We use similar notations as PointNet++ [3] 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 [4], 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 [5] and Leaky ReLU [6] 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 [7], which folds 2D grids into 3D shapes:

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

III. 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 1, local point sets located at different parts of the

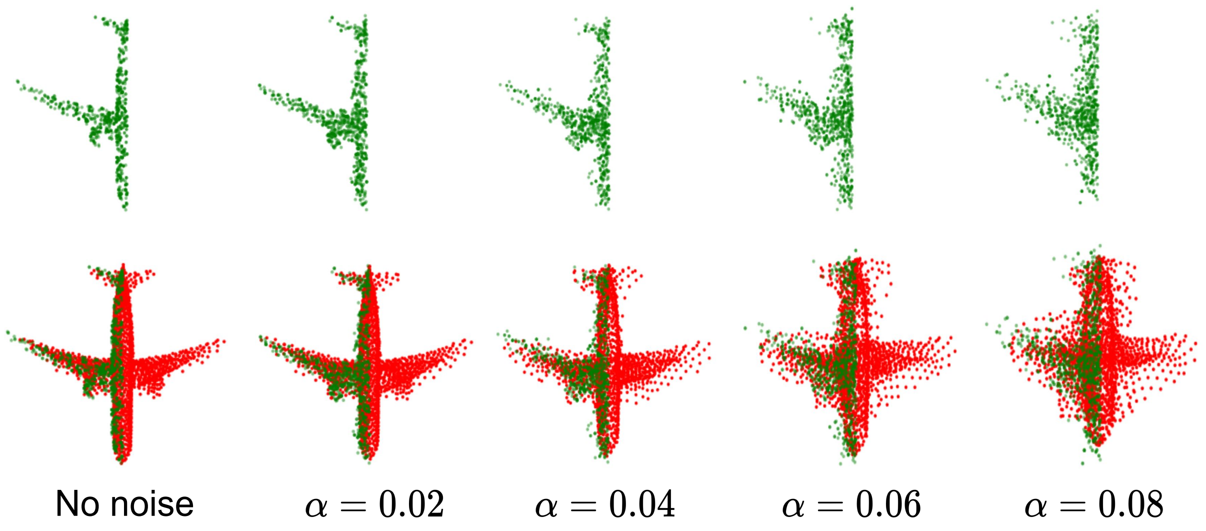

Fig. 2: Results of point clouds completion obtained from the noisy partial observation. Gaussian noise with zero mean is assumed and the standard deviation is indicated at the bottom. Top: input partial observation. Bottom: prediction (red) overlapped with inputs (green).

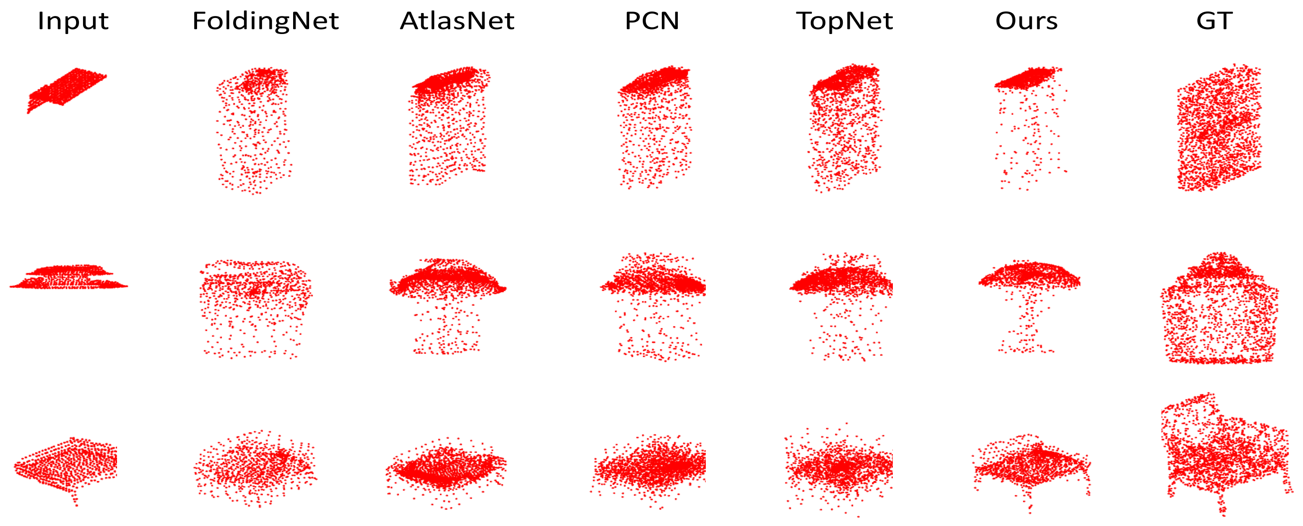

Fig. 3: Failure cases of point clouds completion on the Completion3D dataset.

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.

IV. Visualization of Point Clouds Completion with Noisy Inputs

We visualize the results of point clouds completion with added noise in the Figure 2. 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. It shows that the proposed model tends to maintain input partial shapes and lack the ability to distinguish noise points.

V. Failure Cases on Point Clouds Completion

We show failure cases of point clouds completion on Completion3D in the Figure 3. 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 a mean shape of what they have trained on when difficult partial point clouds are observed. However, the proposed model predicts reasonable complete shapes.

References

[1] M. Fey and J. E. Lenssen, “Fast graph representation learning with PyTorch Geometric,” in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[2] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[3] 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.

[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] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International Conference on Machine Learning*, 2015, pp. 448–456.

[6] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier nonlinearities improve neural network acoustic models,” in *Proc. icml*, vol. 30, 2013, p. 3.

[7] 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.