A Dataset-Dispersion Perspective on Reconstruction Versus Recognition in Single-View 3D Reconstruction Networks

Yefan Zhou
UC Berkeley
yefan0726@berkeley.edu

Yiru Shen
Clemson University
yirus@g.clemson.edu

Yujun Yan
University of Michigan
yujunyan@umich.edu

Chen Feng
New York University
cfeng@nyu.edu

Yaoqing Yang
UC Berkeley
yqyang@berkeley.edu

Abstract

Neural networks (NN) for single-view 3D reconstruction (SVR) have gained in popularity. Recent work points out that for SVR, most cutting-edge NNs have limited performance on reconstructing unseen objects because they rely primarily on recognition (i.e., classification-based methods) rather than shape reconstruction. To understand this issue in depth, we provide a systematic study on when and why NNs prefer recognition to reconstruction and vice versa. Our finding shows that a leading factor in determining recognition versus reconstruction is how "dispersed" the training data is. Thus, we introduce the dispersion score*, a new data-driven metric, to quantify this leading factor and study its effect on NNs. We hypothesize that NNs are biased toward recognition when training images are more dispersed and training shapes are less dispersed. Our hypothesis is supported and the dispersion score is proved effective through our experiments on synthetic and benchmark datasets. We show that the proposed metric is a principal way to analyze reconstruction quality and provides novel information in addition to the conventional reconstruction score. We have open-sourced our code.*[1]

1. Introduction

Using deep learning (DL) for single-view 3D reconstructions (SVR) is our main focus. Numerous recent publications presented innovative neural network (NN) designs to advance the state-of-the-art in SVR [6, 9, 15, 17, 20, 21, 23–25, 27, 28]. Their primary focus is on improving the quality of reconstruction on benchmark datasets, which is measured by reconstruction metrics, such as Chamfer distance (CD) [6], Earth Mover Distance [6], mIoU [5], and F-score [22].

[1] https://github.com/YefanZhou/dispersion-score

While several studies have proposed improved methods to reconstruct shapes, only few focus on the underlying mechanisms of NNs in SVR and whether the actual mechanisms meet the designers' expectations.

Several studies [19, 22] have shown that cutting-edge DL models in SVR primarily perform *recognition* rather than reconstruction. In other words, NNs tend to find a shortcut to solve SVR by using the easy-to-learn classification-based approaches, e.g., by implicitly grouping the training shapes into clusters and memorizing only the mean shapes of these clusters. This mean-shape-based approach is fundamentally different from our intuition of 3D reconstruction.

More importantly, their finding shows that NN has limited performance on reconstructing novel objects when it relies more on recognition, rendering the question of "do NNs perform recognition or reconstruction". Furthermore, findings [19, 22] show that the bias toward recognition can arise from properties of the dataset, e.g., some training dataset uses the *object-centered (OC) coordinate* [5, 6, 9, 16]. In the OC coordinate, the 3D shapes of objects are aligned to the same orientation. In the other *viewer-centered (VC) coordinate*, the 3D shapes are aligned to randomly sampled input viewpoints. In particular, they observe that using OC can make NNs more biased toward recognition than VC. Although these examinations provide valuable insights into how NNs perform SVR, the answer to the question depends on a multitude of factors. Determining whether NNs should perform in OC or VC is not enough to resolve the question.

To address the issue of NNs' bias toward recognition in SVR, we provide a systematic study on recognition versus reconstruction. Our investigation leads to a comprehensive evaluation metric (*dispersion score* or DS) and applicable experiment procedures to improve SVR. In particular, we show that DS can diagnose the trained model's bias toward recognition (in Section 5.3). We also illustrate that the use of more dispersed training shapes can improve reconstruc-

tion as shown by CD and DS (in Section 5.4). Specifically, our main contributions provide answers to the following questions.

What causes NNs to prefer recognition in SVR? In our experiments, we showed that whether NNs would perform reconstruction or recognition depends on if the training data is "dispersed" or "clustered". For example, we hypothesize that OC makes NNs perform recognition because aligning training objects to a common orientation makes the training shapes more clustered. The clustered training shapes can bias NNs toward using recognition-based approaches.[2] Our main claim is that NNs tend toward reconstruction when 3D training shapes are more dispersed. They are prone to perform recognition when the 2D training images are more dispersed.

How can we measure the extent of reconstruction or recognition? We propose a new metric, DS, to measure how dispersed or "unclustered" the data is. A larger score indicates that the data is more dispersed and less clustered, whereas a lower score indicates the opposite. DS is measured from two perspectives: *input DS* on training data and *output DS* on reconstructed shapes (results of test). The input DS is calculated to diagnose the training data and describe its relationship with the corresponding trained models. The output DS is calculated to measure whether the trained models tend to perform reconstruction or recognition. We notice that measuring the DS of reconstructed shapes (i.e., output DS) can indicate whether NNs are biased toward recognition because the output shapes tend to form clusters when the NNs rely on using mean shapes to reconstruct.

Finally, it is worth noting that the question of recognition versus reconstruction is not equivalent to memorization versus generalization. The latter has a clear definition, e.g., Eqn. (1) of [7]. The question of recognition versus reconstruction currently does not have a rigorous definition. We provide the first metric to quantify recognition versus reconstruction. However, the DS is not necessarily a one-size-fits-all statistic that can distinguish recognition from reconstruction. We know that existing reconstruction metrics like CD, which measures a single shape's quality, cannot tell whether the reconstruction uses the mean shape because quantifying the mean shape requires measuring more than one shape. Thus, the proposed DS provides novel information in addition to the conventional reconstruction score when assessing SVR models.

2. Related Work

Single-view 3D reconstruction There have been lots of studies on DL-based SVR using various 3D representations, including voxels [5], point clouds [6, 9, 28], meshes [8, 24], and signed distance fields (SDF) [16, 26]. These techniques have been proven efficient in improving the quality of shape reconstruction, measured by similarity metrics such as CD, Earth Mover Distance [6], mIoU [5], and F-score [22]. A critical difference between these work and ours is that they make a single predicted shape closer to the ground truth shape by neglecting if the NNs use recognition-based or reconstruction-based schemes, which requires the knowledge obtained from the whole dataset.

Reconstruction vs Recognition Recently, a few studies advocate a rethinking of how NNs perform SVR tasks. In particular, the mechanism of SVR is hypothesized to be a combination between reconstruction and recognition [19, 22]. For example, [19] proposes that the commonly used shape representation and object-centered coordinate make NNs place more importance on recognizing the object category (or cluster) and thus encourage memorizing the object shape. This hypothesis is supported by the qualitative results that trained NN models sometimes predict a shape in an entirely different object category than the input image, which is conjectured to be caused by a classification error. Further, [22] shows that state-of-art NNs for SVR tasks rely predominantly on recognition instead of reconstruction. The claim is supported by observing that NNs have similar reconstruction performance with recognition-based methods measured by the mIoU score. Although prior work points out some factors that could bias NNs toward recognition, their findings are limited to special issues like shape coordinate representation. In our work, we provide a more systematic view of this problem and give operational ways to guide NNs toward reconstruction.

Choice of The Coordinate Representation The conventional setting for SVR tasks is to predict output shapes in OC coordinate [5, 6, 9, 16, 26]. However, VC coordinate is recommended to alleviate the bias toward recognition [22] and improve the generalization ability to reconstruct unseen object classes [19]. In this work, we study the impact of the two coordinate representations on the DS of datasets and trained models.

3. Definition and Main Claim

3.1. Single-view 3D Reconstruction

We consider the problem of SVR using NNs. The input $I \in \mathbb{R}^{W \times H}$ is a 2D image with width W and height H. The output, denoted as S, is a 3D point cloud $\in \mathbb{R}^{N \times 3}$. We only consider point-cloud-based shape representation in this work. A NN model f is trained to reconstruct the shape S from the input image I, by minimizing the empirical loss defined for a certain loss function l:

$$\min_{f} \sum_{i=0}^{n-1} l(f(I_i), S_i). \tag{1}$$

[2] In the Appendix A, we provide further details of OC and VC, and more experiments on their difference.

3.2. Recognition vs. Reconstruction

Recognition and reconstruction are two modes that NNs can perform in the SVR tasks. The basic mechanisms of these two are outlined as the following:

Recognition A recognition-based model reconstructs shapes in two steps. First, during training, the model partitions the training shapes into clusters based on shape similarity and memorizes the mean shape of each cluster. Then, during testing, the model classifies the input test image into one of the clusters and retrieves the corresponding mean shape as the output. In this case, the reconstructed shapes are highly clustered because different input images could be classified into one single mean shape.

Reconstruction A reconstruction-based model directly generates the 3D reconstruction rather than using any cluster or semantic information. In this case, the reconstructed shapes are dispersed because the feature of each shape corresponds to the low-level image cues.

The two mechanisms described above are distinctive but not disjoint. It is known that a trained NN in practice combines these two to perform SVR. The investigation of recognition versus reconstruction is different from memorization versus generalization of NN in SVR [2]. The latter focuses on reducing the generalization gap between training and test, while our work only studies the working mechanism of NNs for SVR.

3.3. Dispersion Score Metric

We define the dispersion score (DS) to measure how dispersed the data is. The metric is defined based on the classical notion of *clustering inertia* [4].

Given a dataset $D = \{x_i\}_{i=0}^{N-1}$ and a distance function $d(x, y)$, we first determine *clustering* of the dataset by using the K-medoids algorithm [10]. We provide ablation studies on different clustering methodologies in Appendix B. Given the number of clusters n, the clustering result is denoted as $C_n(\cdot)$, where for each sample $x_i \in D$, K-medoids gives the cluster label $C_n(x_i)$. Ctr_i denotes the centroid of the cluster that contains the sample x_i. Then, the inertia I of the dataset D partitioned by $C_n(\cdot)$ is defined as:

$$I_{C_n}(D) = \sum_{i=0}^{N-1} d(x_i, Ctr_i). \tag{2}$$

The DS, defined using the inertia, is given by:

$$DS(D) = \frac{I_{C_n}(D)}{N}. \tag{3}$$

DS measures the average distance of each sample to its assigned cluster centroid. Thus, with a larger DS, the sample is further away from its cluster centroid and the dataset is more dispersed. For example, if the NN performs pure recognition, each reconstructed shape is equal to the corresponding cluster's mean shape (cluster centroid). In this case, the DS of the reconstructed shapes equals 0, which is the limit of pure recognition.

The K-medoids algorithm requires assigning the number of clusters. We automatically determine this hyperparameter using the "Kneedle" method [18]. The detail is provided in Appendix C. When evaluating DS in SVR tasks, we need to define the distance function $d(x, y)$ to measure pairwise distance between data samples. For 3D point cloud data, we define $d(x, y)$ as CD which measures the distance between two point sets. For the 2D image data, we define $d(x, y)$ as the *feature reconstruction loss* which compares image contents in a high dimensional feature space [13].

3.4. Dispersion Relationship Hypothesis

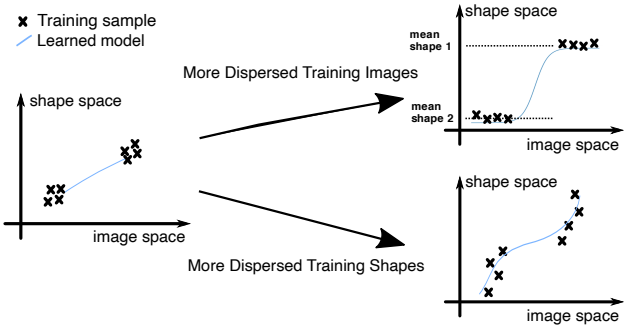


Figure 1: Caricature of the two approaches to change training datasets' DS: using more dispersed training images makes the reconstructed shapes more clustered, whereas using more dispersed training shapes makes the reconstructed shapes more dispersed.

In this paper, we experimentally demonstrate that the bias of trained NNs toward either recognition or reconstruction depends on whether the training dataset is clustered or dispersed. Specifically, we consider two ways to change datasets' DS: *more dispersed training images* and *more dispersed training shapes*, as illustrated in Figure 1. We will experimentally demonstrate the following claim.

(Main claim) *In SVR, the more dispersed training images make NNs biased toward recognition, whereas the more dispersed training shapes guide NNs to use reconstruction.*

The claim is motivated by prior SVR work. First, [22] proposes to use VC instead of OC. Training shapes in VC are more dispersed than OC, while training images in both cases are the same. Second, it is common to use image augmentations to enhance SVR [9, 16], in which training images become more dispersed while training shapes remain unchanged.

We illustrate intuition behind the main claim in Figure 1. By making training images more dispersed with the clus-

tered training shapes, we make the trained model exhibit a higher tendency toward recognition, i.e., the shape predictions concentrate on the mean shapes and are highly clustered. The clustered shape predictions are illustrated in Figure 1 as the intersections between the two dashed lines and the y-axis. On the other hand, making training shapes more dispersed in the shape space guides the model to learn more dispersed shape data. Thus, the NNs learn to reconstruct more dispersed shapes and rely less on mean shapes.

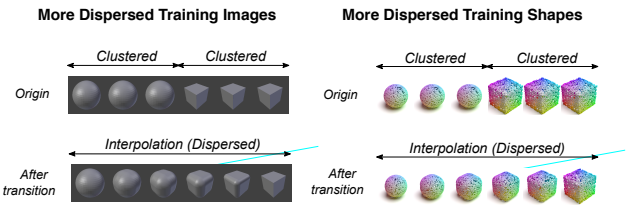
4. Experiments on Synthetic Dataset

In this section, we verify our claim on synthetic datasets. We first describe the designs of the synthetic datasets, which correspond to the two transitions proposed in Section 3.4, specifically, more dispersed training images and shapes. We then show the effects of the two transitions on NNs' tendency toward recognition or reconstruction by analyzing both *distance matrices* and input/output DS. The definition of distance matrices is in Section 4.2.

4.1. Dataset

We create and split a synthetic base dataset into training and test sets. By sampling instances from the training set of the base dataset, we generate two groups of sub-trainsets representing the varying DS for training images and shapes.

Synthetic Shape Generation The base dataset is generated by interpolating between a cube and a sphere of a similar size. We use Blender [11] to implement the interpolation, and more details are in Appendix D. The interpolation generates 1000 intermediate shapes. Given each intermediate shape in mesh format, we render an image from the isometric view and sample a point cloud consisting of 2500 points. We use the image and the point cloud to comprise an image-shape pair as a data sample.



(a) (b)

Figure 2: The illustration of the two transitions implemented by toy examples. (a) Interpolating more images between sphere and cube images makes training images more dispersed. (b) Interpolating more shapes between sphere and cube shapes makes training shapes more dispersed.

Dataset Composition We use the same test set generated from the base dataset for all the experiments. We produce the training sets in different experiments to create scenarios with varying dispersed training images or shapes. In particular, we build sub-trainsets with uncorrelated image and shape DS by sampling images and shapes independently from the base training set.

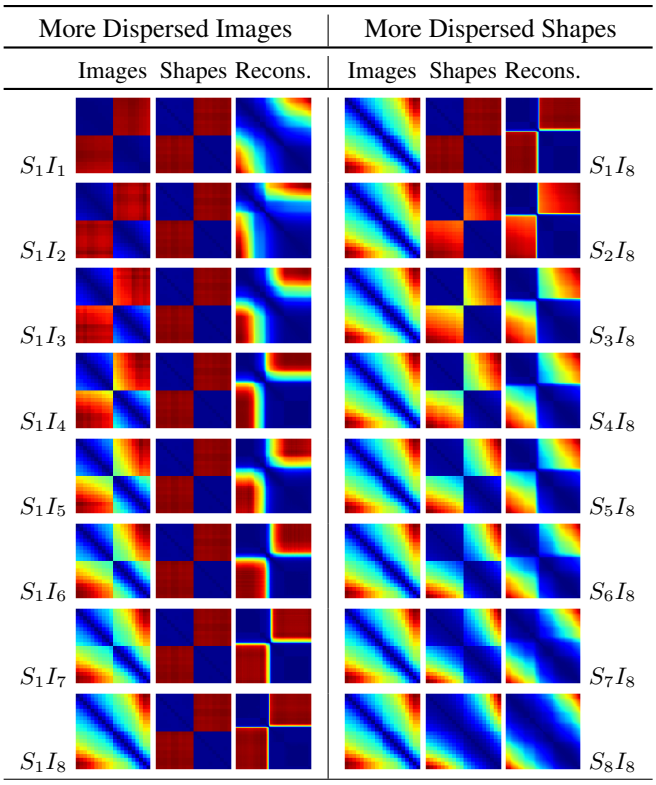


Figure 3: Visualization of the distance matrices from experiments on sub-trainsets S_1I_i and S_jI_8. Blue represents small distance and red represents large distance. *Images*: training images. *Shapes*: training shapes. *Recons*: trained models' reconstructed shapes on an identical test set. *More Dispersed Images (Left Columns)*: Reconstructed shapes become more clustered when training images are more dispersed. *More Dispersed Shapes (Right Columns)*: Reconstructed shapes become more dispersed when training shapes are more dispersed.

Figure 2a shows the transition with more dispersed training images, in which the training images are highly clustered initially and more dispersed after the transition. We build a group of 8 sub-trainsets to gradually increase image DS by interpolating between the two ends of this transition. This procedure gives the first group of the 8 sub-trainsets. The corresponding training shape sets are identical across the 8 sub-trainsets and are specifically designed to be highly clustered. The design aims to approximate the most common scenario when training shapes are clustered, e.g. when the OC coordinates are used.

Figure 2b shows the other transition with more dispersed training shapes. The training shapes change from being

clustered to being dispersed. Similarly, we build a group of 8 sub-trainsets to gradually increase shape DS by interpolating between the two ends of this transition. Also, note that the corresponding training image sets are the same across the 8 sub-trainsets and are kept highly dispersed to approximate the most common scenario in ShapeNet that training images have a large variety of lighting, viewpoint, texture.

We use S_1I_i and S_jI_8 $(i, j = 1, 2, \ldots, 8)$ to symbolize these two groups of sub-trainsets. S represents training shapes while I represents training images. The subscript i and j represent the DS's order of training images and training shapes, respectively. Thus, S_1I_i $(i = 1, 2, \ldots, 8)$ represents the sub-trainsets with the least dispersed training shapes and gradually more dispersed training images, corresponding to the scenario in Figure 2a. Similarly, S_jI_8 $(j = 1, 2, \ldots, 8)$ represents the sub-trainsets with the most dispersed training images and gradually more dispersed training shapes, corresponding to the scenario in Figure 2b.

4.2. Implementation Details

We adopt AtlasNet-Sphere [9] as the baseline model. For two groups of 8 sub-trainsets, we train 16 models separately using training protocol detailed in Appendix E and evaluate the trained models from the last epoch. We compute the distance matrices of the 16 reconstructed shape sets and corresponding training data, visualizing them in Figure 3. The distance functions for different types of data are defined in Section 3.3. We further evaluate the input/output DS of the 16 sub-trainsets and the trained models. The results are reported in Figure 4. The cluster-number hyperparameter of input/output DS is set to be 2.

4.3. More Dispersed Training Images

We now show results to support the main claim in Section 3.4. Note that one part of the claim is that NNs tend toward recognition when training images are more dispersed. We use two evaluation methods to analyze our experiments: distance matrices (shown in Figure 3) and DS (shown in Figure 4).

We first analyze Figure 3 to show how the distance matrices support the claim. Each subfigure in Figure 3 represents a distance matrix measured on one of the datasets or reconstructed shape sets. The reconstructed shape sets are generated at test time. In this subsection, we parse the left three columns of Figure 3.

We measure the distance matrices of training images (first column), training shapes (second column), and reconstructed shapes on the test set (third column). Now, look at the first row marked by S_1I_1. The two distance matrices in the first and second columns, titled "Images" and "Shapes" respectively, show sudden color mutation from blue to red, indicating training images and shapes are highly clustered. The distance matrix in the column titled "Recons" shows a continuous color change from blue (low value) to red (high value), meaning the reconstructed shapes are dispersed. Then, we can analyze the following rows marked by S_1I_i $(i = 2, \ldots, 8)$ similarly. The distance matrices on the "Recons" column show increasingly clustered patterns, while the distance matrices under the "Images" column show increasingly dispersed patterns. It indicates that the reconstructed shapes of NNs become more clustered, which supports our claim that NNs tend toward recognition when training images are more dispersed.

Next, we look at the values of DS for training images, training shapes, and reconstructed shapes, shown in Figure 4a, 4b, and 4c, respectively. Figure 4a shows that the input DS of the training image sets gradually increases, indicating that the training image sets become more dispersed. Note that this increasingly dispersed pattern matches the first column in Figure 3. Then, Figure 4b shows that the training shapes remain unchanged, matching the second column in Figure 3. Finally, Figure 4c shows that the output DS of reconstructed shapes gradually decreases, matching the increasingly clustered color patterns shown in the third column of Figure 3.

Thus, both the distance matrices (shown in Figure 3) and the DS trends (shown in Figure 4) show that NNs prone to perform recognition and predict more clustered shapes when training images become more dispersed.

4.4. More Dispersed Training Shapes

The other part of our main claim is that NNs tend toward reconstruction when training shapes become more dispersed. We conduct the same analysis as the previous subsection using both distance matrices and DS.

For distance matrices, see the right three columns in Figure 3. From top to bottom, the column titled "Shapes" shows increasingly dispersed patterns while the column titled "Images" remains dispersed. It indicates a list of training datasets of more dispersed shapes and consistently dispersed images. The column titled "Recons" shows more dispersed color patterns, meaning that the reconstructed shapes become dispersed. Thus, more dispersed training shapes make the reconstructed shapes more dispersed, indicating that NNs tend toward reconstruction.

The results are again verified by DS evaluation, as shown in Figure 4d, 4e, and 4f. First, Figure 4d shows that the input DS of training images remains unchanged. Second, Figure 4e shows that the input DS of training shapes gradually increases. Finally, Figure 4f shows that the output DS of the reconstructed shapes gradually increases. These trends all match the results shown in Figure 3.

Therefore, both of the two evaluation methods support the claim that NNs lean toward reconstruction and predict more dispersed shapes when training shapes also become more dispersed.

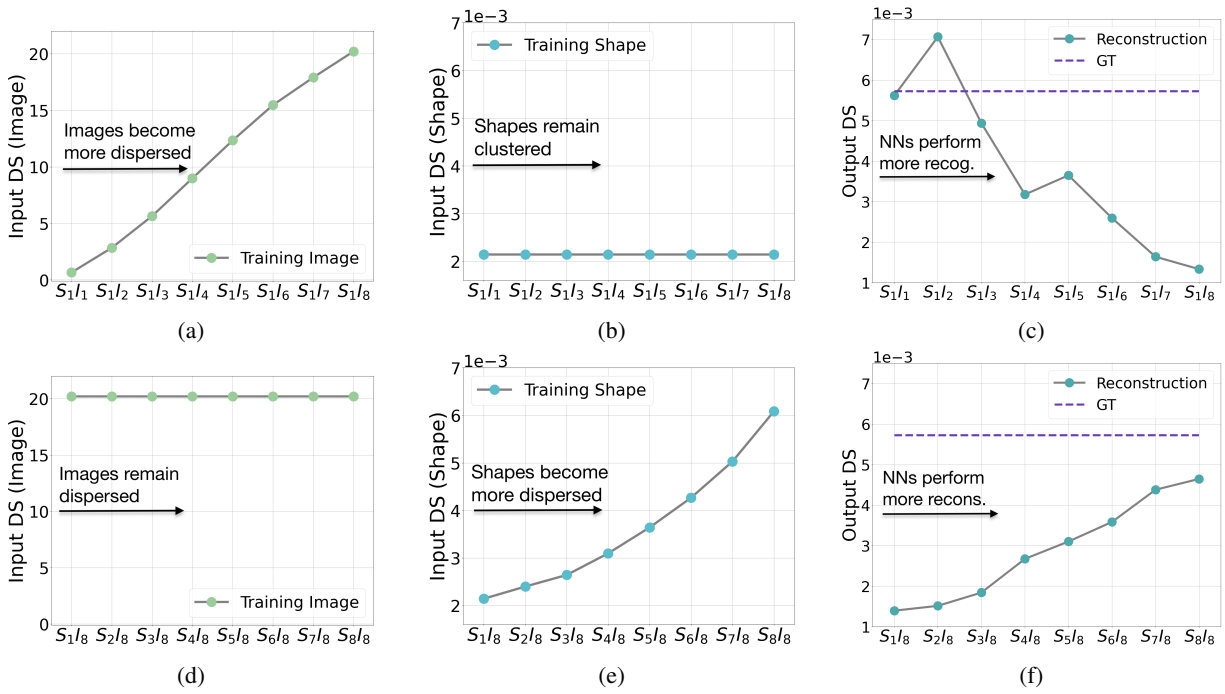


Figure 4: Input/Output DS of synthetic datasets and trained models. (a-c) **More dispersed images** make NNs tend toward recognition as the reconstructed shapes become less dispersed when training images become more dispersed. (d-f) **More dispersed shapes** make NNs tend toward reconstruction as the reconstructed shapes become more dispersed when training shapes become more dispersed. GT: ground-truth shapes of test set.

5. Experiments on ShapeNet

In this section, We further verify our main claim by conducting experiments on the commonly used benchmark dataset ShapeNet [3]. We show how varying levels of dispersed images and shapes affect the tendency of NNs to perform reconstruction or recognition. Due to space limitations, we focus on encoder-decoder-based NNs, including PSGN [6], FoldingNet [28], and AtlasNet [9]. We conduct additional experiments on SDF-based NNs in Appendix F.

5.1. Dataset

ShapeNet We conduct experiments on ShapeNetCore consisting of 3D models in 13 object categories [3]. We use the train/test split in [5] and use the point cloud data provided by AtlasNet [9]. In experiment 5.3, we render new image datasets using the method of [26] to control the rendering viewpoints of training images. Both OC and VC coordinates are investigated. In experiments 5.4, we use images rendered by [5]. In this dataset, each 3D model has been rendered 24 images of random views. We use one fixed view among 24 views in the training/test set in 5.4 and investigate the impact of using more views per shape in 5.5.

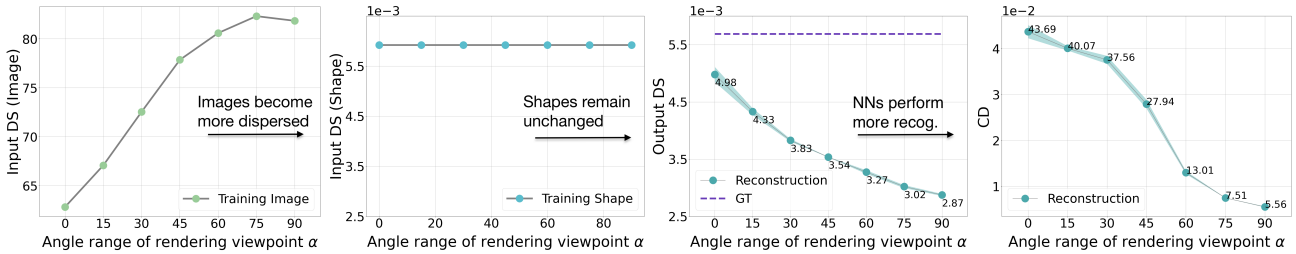
5.2. Implementation Details

For experiments in 5.3 and 5.5, we adopt AtlasNet-Sphere [9] as the baseline. In 5.4, we use multiple SVR models. The details of model implementation and training are provided in Appendix E. We evaluate the trained model from the last epoch. We run the model on each dataset using three random seeds and report the mean and standard deviation for evaluation. Point clouds are used as shape representation. Each point cloud includes 2500 points. The cluster-number hyperparameter of input/output DS is set to be 500.

5.3. More Dispersed Training Images

We study the transition more dispersed training images. The image set is generated to be more dispersed while the shape set remains clustered.

Experiment Design We render a list of new image datasets from the training shapes of ShapeNet and increase the angle range of the rendering viewpoint. The angle range is denoted by α, and the unit is degree. We build seven rendering datasets in this way. For each of them, we only render a single image for each shape. During rendering, the θ_{az} of viewpoint is randomly sampled from $-\alpha$ to α and θ_{el} is sampled from 20 to 30 degree. The θ_{az} and θ_{el} are azimuth angle and elevation angle of viewpoint, respectively. α is



(a) (b) (c) (d)

Figure 5: DS measure in OC when the training images are rendered by increasing viewpoint angle range α. The unit of α is degree. (a) (b) From left to right, training images become more dispersed and training shapes remain unchanged. (c) Reconstructed shapes become less dispersed. (d) CD scores of reconstructed shapes become smaller. The smaller, the better.

selected as 0, 15, 30, 45, 60, 75, 90 for the seven different datasets. As α increases, training images become more dispersed. We only use shapes in OC coordinate so that the training shapes remain clustered. During testing, the input images of the test set are rendered using an α value equal to 90.

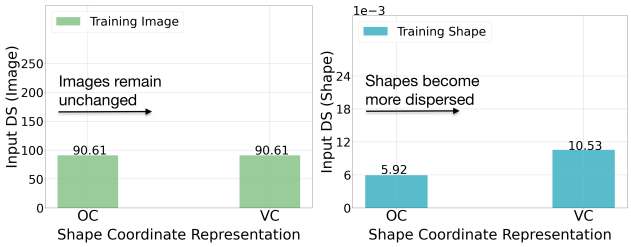
Results We show how NNs perform when the training images become more dispersed. First, Figure 5a and 5b show that our approach makes training images more dispersed while maintaining the DS of training shapes. Then, Figure 5c shows that the trained NNs tend more towards recognition as the value of output DS decreases. This trend matches our main claim that more dispersed training images make the output shapes tend towards recognition.

However, Figure 5d shows that NNs trained on more dispersed images have improved reconstruction score measured by the CD. This improvement could result from augmenting the training dataset or the case where the distribution of images in the training set becomes closer to that of the test set under this transition. Thus, if only looking at the CD measure, one may believe that adding more training images can lead to improved 3D reconstruction quality but may neglect the confounding factor that the output shapes become more clustered. This finding shows that while more dispersed images can potentially improve the CD score, they also incline NN to reconstruct more clustered shapes. In other words, the improved CD score is not sufficient to capture whether the output shapes become more clustered or not. Therefore, we illustrate that the proposed DS provides novel information in addition to the conventional reconstruction score.

5.4. More Dispersed Training Shapes

We study what happens when we use gradually more dispersed training shapes while maintaining the training images dispersed.

Experiment Design Two coordinate representations OC and VC are used here to change the input DS of the ShapeNet dataset. As mentioned in Section 1, shape sets



(a) (b)

Figure 6: Comparing Input DS of OC and VC. (a) Training images remain unchanged. (b) Training shapes become more dispersed.

in VC are more dispersed than those in OC. We use the same training images for both two datasets to ensure that the DS of training images remains unchanged. We train four different SVR models: PSGN [6], AtlasNet-Sphere [9], AtlasNet-25 [9], and FoldingNet [28]. We measure these models in both OC and VC and use the output DS to see if the predicted shapes are more or less dispersed. However, note that the output shapes are not in the same coordinates and are not directly comparable if we change the coordinates. Thus, during the calculation of output DS, we transform the predicted shapes back to OC coordinates to keep the same output configuration for a fair comparison on output DS.

Results We now present our results to demonstrate that the transition with more dispersed shapes guides NNs towards more reconstruction. See Figure 6 for the input DS and Table 1 for the output DS. From Figure 6, we see that the transition makes training shapes more dispersed while letting training images remain unchanged. Then, from Table 1, we see that models trained in the VC coordinate have more dispersed shape predictions than OC, as the output DS values of all the models in VC are larger than that in OC. Besides, in Table 1, we also measure the CD, and we see that all the VC models outperform OC models. Based on the two observations, we can conclude that more dispersed train-

	Output DS		CD	
	OC	VC	OC	VC
PSGN	1.63 ± 0.07	$\mathbf{2.47 \pm 0.00}$	6.60 ± 0.12	$\mathbf{5.90 \pm 0.15}$
FoldingNet	2.39 ± 0.04	$\mathbf{3.34 \pm 0.05}$	7.26 ± 0.08	$\mathbf{5.84 \pm 0.10}$
AtlasNet-Sph.	2.83 ± 0.00	$\mathbf{3.60 \pm 0.02}$	7.12 ± 0.08	$\mathbf{5.40 \pm 0.02}$
AtlasNet-25	2.84 ± 0.01	$\mathbf{3.55 \pm 0.01}$	6.59 ± 0.07	$\mathbf{5.06 \pm 0.02}$
GT	5.68	5.68	-	-

Table 1: Evaluation (mean±stdev) of models trained in OC (*Left*) and VC (*Right*) coordinates. Metrics are output DS ($\times$ 0.001, $\uparrow$), CD ($\times$ 0.001, $\downarrow$). NNs trained in VC predict more dispersed and better reconstructed shapes than NNs in OC.

ing shapes encourage NNs to use more reconstruction than recognition as the underlying mechanism to perform. At the same time, more dispersed training shapes also improve reconstruction performance measured by CD.

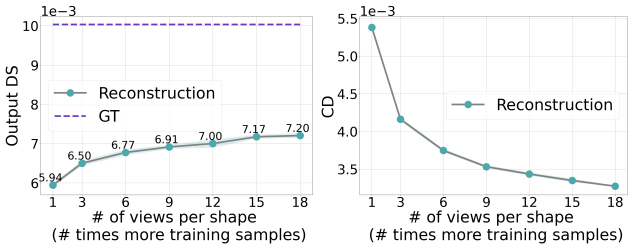
Finally, from the results of Section 5.3 and 5.4, we see that both more dispersed training images and more dispersed training shapes can lead to improved reconstruction scores. However, more dispersed training images actually let the NNs prefer recognition to reconstruction. Such results again show that our new way of measuring the dispersion of output shapes provides novel information on assessing the 3D reconstruction quality.

5.5. More Training Samples

We investigate whether more training samples can guide NNs to perform more reconstruction in SVR. Note that in Section 5.3 and 5.4, we only make training shapes more dispersed or only make training images more dispersed, to show how the output DS changes with each covariate. In this subsection, we conduct this additional experiment to change training shapes and images simultaneously because it is often practically convenient to do so, e.g., by adding more samples.

Experiment Design We obtain more training samples using more rendered images that have been given in [5]. The conventional training protocol in [5, 9] is to use one view of the image among 24 views per shape for each epoch. However, in this experiment, we use more views per shape for each epoch, which ranges from 1 to 18. We use the VC coordinate, and hence shapes are rotated based on the input viewpoint of the rendered images. Thus, using more views per shape is equivalent to using more training images and more training shapes obtained by performing rotations in the 3D space. Also, the amount of training data is linear in the number of views per shape. We adopt AtlasNet-Sphere [9] as the baseline model and use the same protocol in Section 5.2.

Results The results are reported in Figure 7. First, both output DS and CD are shown to improve with more training samples. More specifically, the increased output DS indicates that more training samples guide NNs to perform more reconstruction. And the decreased CD score indicates that more training samples improve the reconstruction quality. Second, Figure 7a shows a noticeable gap between GT (0.01) and the limit of the improved output DS (0.0072) of NNs trained on 18$\times$ more data samples. This indicates that it is challenging to further improve NNs to perform reconstruction based on simply augmenting the public dataset.



(a) (b)

Figure 7: Evaluation of models trained on more data samples in VC. The amount of training data samples is linear in the number of views per shape. Although more training shapes make NNs prefer reconstruction, the capability of NNs to perform reconstruction is still limited, even using 18$\times$ training data. (a) Output DS ($\uparrow$). (b) CD ($\downarrow$).

6. Conclusion

In this paper, we study the underlying mechanisms of NNs in SVR tasks. First, we show that NNs can be disposed towards recognition or reconstruction depending on how dispersed the training data is. We propose a metric called DS to quantify this relationship. We show that both of the two experiment procedures, i.e., using more dispersed training images and shapes, can improve conventional reconstruction scores such as CD. However, the DS measure shows that the former (training images) leads NNs to prefer recognition rather than reconstruction while the latter (training shapes) leads NNs to perform more reconstruction. Thus, the proposed DS provides novel information on how NNs perform SVR tasks. We suggest measuring the DS in conjunction with conventional reconstruction scores when assessing trained NNs in SVR tasks. More studies on other DL techniques, including data augmentation and network architectures, would be necessary to make NNs perform reconstruction instead of recognition.

Acknowledgments

This research is partially supported by NSF Future Manufacturing program under EEC-2036870.

References

[1] David Arthur and Sergei Vassilvitskii. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, page 1027–1035, USA, 2007. Society for Industrial and Applied Mathematics. 4331

[2] Miguel Angel Bautista, Walter Talbott, Shuangfei Zhai, Nitish Srivastava, and Joshua M Susskind. On the generalization of learning-based 3d reconstruction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2180–2189, 2021. 4323

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 4326, 4331, 4332

[4] C. Chinrungrueng and C. H. Sequin. Optimal adaptive k-means algorithm with dynamic adjustment of learning rate. *IEEE Transactions on Neural Networks*, 6(1):157–169, 1995. 4323

[5] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016. 4321, 4322, 4326, 4328, 4331

[6] Haoqiang Fan, Hao Su, and Leonidas 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*, pages 605–613, 2017. 4321, 4322, 4326, 4327, 4333

[7] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. *arXiv preprint arXiv:2008.03703*, 2020. 4322

[8] Georgia Gkioxari, Jitendra Malik, and Justin J Johnson. Mesh r-cnn. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9784–9794, 2019. 4322

[9] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018. 4321, 4322, 4323, 4325, 4326, 4327, 4328, 4333

[10] Francesco Gullo, Giovanni Ponti, and Andrea Tagarelli. Clustering uncertain data via k-medoids. In Sergio Greco and Thomas Lukasiewicz, editors, *Scalable Uncertainty Management*, pages 229–242, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. 4323

[11] Roland Hess. *Blender Foundations: The Essential Guide to Learning Blender 2.6*. Focal Press, 2010. 4324

[12] Geoffrey Hinton and Sam Roweis. Stochastic neighbor embedding. In *Proceedings of the 15th International Conference on Neural Information Processing Systems*, NIPS'02, page 857–864, Cambridge, MA, USA, 2002. MIT Press. 4331

[13] J. Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 4323

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4333

[15] Chun-Liang Li, Manzil Zaheer, Yang Zhang, Barnabas Poczos, and Ruslan Salakhutdinov. Point cloud gan. *arXiv preprint arXiv:1810.05795*, 2018. 4321

[16] Lars M. Mescheder, Michael Oechsle, M. Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4455–4465, 2019. 4321, 4322, 4323, 4333, 4334

[17] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 4321

[18] Ville Satopaa, Jeannie Albrecht, David Irwin, and Barath Raghavan. Finding a "kneedle" in a haystack: Detecting knee points in system behavior. In *2011 31st International Conference on Distributed Computing Systems Workshops*, pages 166–171, 2011. 4323, 4332

[19] Daeyun Shin, Charless C. Fowlkes, and Derek Hoiem. Pixels, voxels, and views: A study of shape representations for single view 3d object shape prediction. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3061–3069, 2018. 4321, 4322

[20] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2018. 4321

[21] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2088–2096, 2017. 4321

[22] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3405–3414, 2019. 4321, 4322, 4323

[23] Shubham Tulsiani, Tinghui Zhou, Alexei A Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2626–2634, 2017. 4321

[24] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018. 4322

[25] Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum. Marrnet: 3d shape reconstruction via 2.5 d sketches. In *Advances in neural information processing systems*, pages 540–550, 2017. 4321

[26] Qiangeng Xu, Weiyue Wang, D. Ceylan, R. Mech, and U. Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. In *NeurIPS*, 2019.

4322, 4326

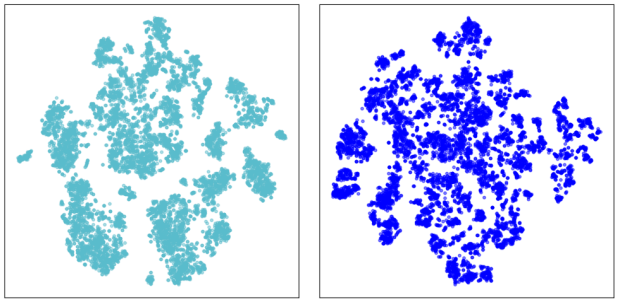
[27] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *Advances in neural information processing systems*, pages 1696–1704, 2016. 4321

[28] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215, 2018. 4321, 4322, 4326, 4327, 4333

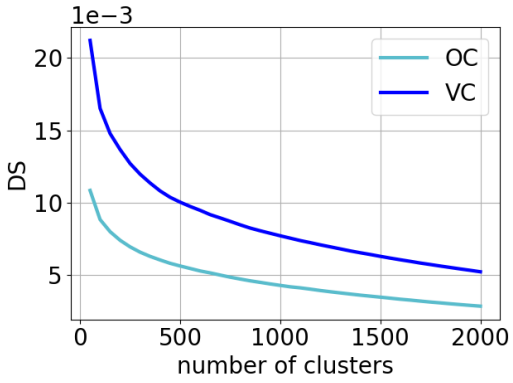
Appendices

A. Choice of The Coordinate Representation

First, we provide the definition of object-centered (OC) and viewer-centered (VC) coordinates. Then, we provide quantitative and qualitative results to show the difference between shapes in the two coordinate representations.



(a) (b)



(c)

Figure 8: (a) t-SNE of distance matrix of shapes in OC coordinate. (b) t-SNE of distance matrix of shapes in VC coordinate. (c) The DS of shapes in OC and VC with varying number of clusters.

As shown in Figure 9, we visualize some shapes in ShapeNet both in OC and VC coordinates. Given a single RGB image as input, we want to predict the 3D shape of the object from which the image is taken. In the OC coordinate, the shapes are predicted in canonical coordinates specified in the training set. For example, in the ShapeNetCore [3] dataset, the $(\theta_{\text{az}} = 0°, \theta_{\text{el}} = 0°)$ direction corresponds to the commonly agreed front of the object, where θ_{az} and θ_{el} are the azimuth and elevation angle of viewpoint. In the VC coordinate, the NN is supervised to predict a prealigned 3D shape in the input image's reference frame. The image-shape pair ensures that $(\theta_{\text{az}} = 0°, \theta_{\text{el}} = 0°)$ in the output coordinate system always corresponds to the input viewpoint.

We further show the different impacts of the two coordinates on training shapes. The main difference is that shapes in OC are more clustered, while shapes in VC are much more dispersed. We use all the shapes of the ShapeNetCore [3] test set split by [5] and represent them both in OC and VC coordinates. There are 8762 shapes in total. These shapes cover 13 semantic classes, and each of them is represented as a point cloud with 2500 3D points. First, we compute distance matrices of shapes using Chamfer distance as the distance function. Then, we visualize the matrices by t-SNE [12] in Figure 8a and 8b. Comparing Figure 8a with 8b, we see that Figure 8a shows a more clustered pattern, while Figure 8b shows a more dispersed pattern. It indicates that shapes in OC are more clustered than those in VC. Besides, we also measure the DS of shapes. We sweep the number of clusters (NC) from 50 to 2000 with step size 50. The results are shown in Figure 8c. The DS of shapes in VC is clearly larger, indicating that the VC coordinate makes shapes more dispersed.

B. Ablation Study of Clustering Methodology

We study two more common clustering methods, namely hierarchical clustering and affinity propagation (AP), besides the K-medoids method. Figure 10 and Table 2 show the DS obtained with the three clustering methods evaluated on all the main experiments on both synthetic and ShapeNet datasets. The results show consistent trends for different clustering algorithms.

We provide the implementation detail of clustering methods [3]. K-medoids and hierarchical clustering require assigning the NC. The value of NC for the synthetic dataset is 2 and for Shapenet is 500. It is automatically determined by the "Kneedle" method detailed in Appendix C. K-medoids is initialized by the "k-means++" [1]. The linkage criterion of hierarchical clustering is the maximum distances between all observations of the two sets. For affinity propagation, we construct the affinity matrix by normalizing the distance matrix by standard deviation and negative exponential. Then we set the hyperparameter "perference", which controls how many exemplars are used, to be the top q-th percentile of entries in the affinity matrix. For synthetic dataset, $q = 4$. For ShapeNet, $q = 60$.

	K-medoids		Hierarchical		Affinity Propagation	
	OC	VC	OC	VC	OC	VC
PSGN	1.63 ± 0.07	$\mathbf{2.47 \pm 0.00}$	1.84 ± 0.09	$\mathbf{2.72 \pm 0.00}$	2.51 ± 0.13	$\mathbf{3.31 \pm 0.00}$
FoldingNet	2.39 ± 0.04	$\mathbf{3.34 \pm 0.05}$	2.66 ± 0.03	$\mathbf{3.78 \pm 0.07}$	3.55 ± 0.02	$\mathbf{4.53 \pm 0.04}$
AtlasNet-Sph.	2.83 ± 0.00	$\mathbf{3.60 \pm 0.02}$	3.18 ± 0.00	$\mathbf{4.04 \pm 0.04}$	4.21 ± 0.01	$\mathbf{5.04 \pm 0.05}$
AtlasNet-25	2.84 ± 0.01	$\mathbf{3.55 \pm 0.01}$	3.17 ± 0.01	$\mathbf{3.95 \pm 0.03}$	4.20 ± 0.04	$\mathbf{4.96 \pm 0.01}$

Table 2: Output DS (mean±stdev, $\times$ 0.001, $\uparrow$) of models trained on more dispersed training shapes in ShapeNet.

[3] All the methods are implemented based on scikit-learn package.

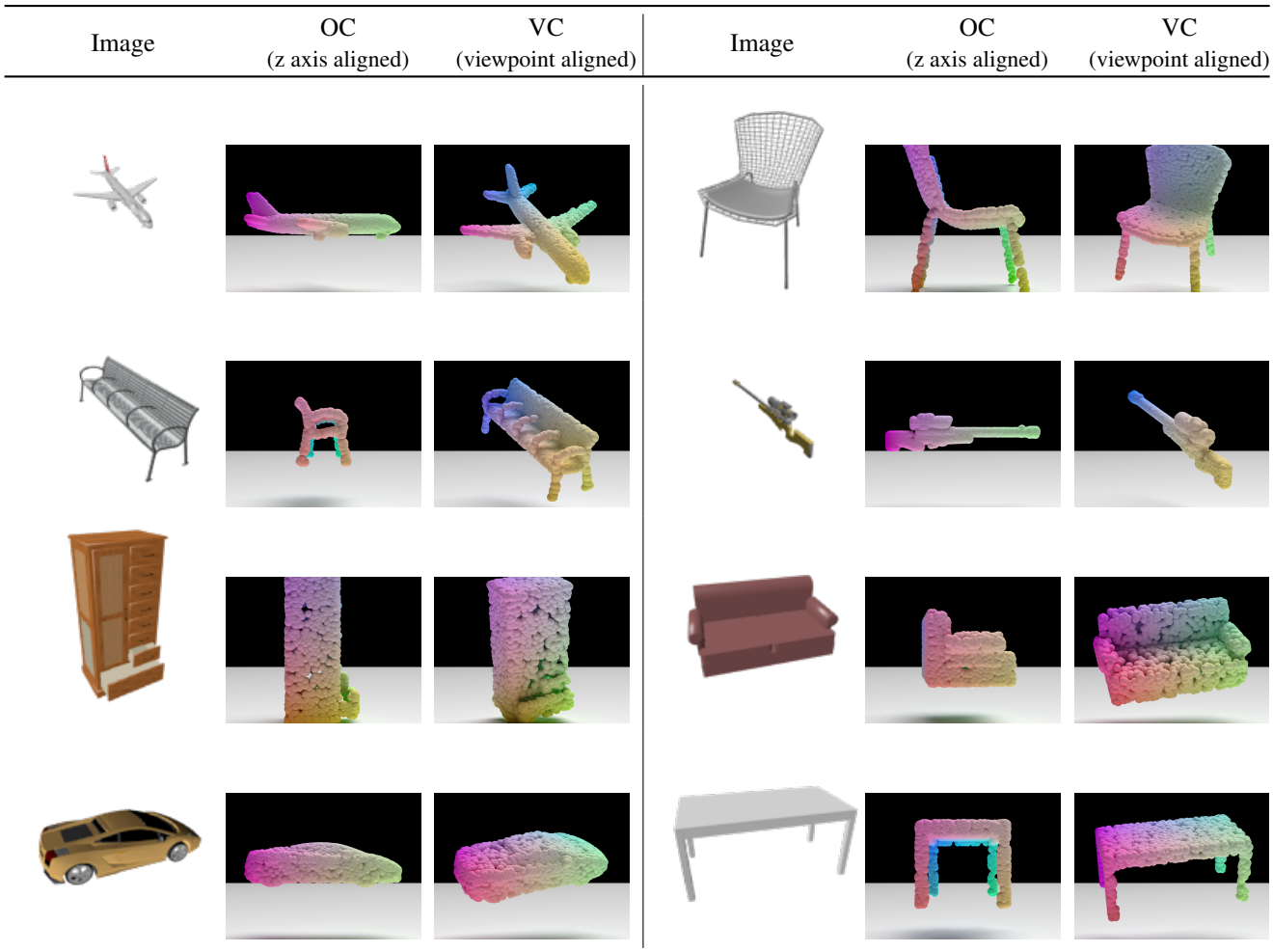
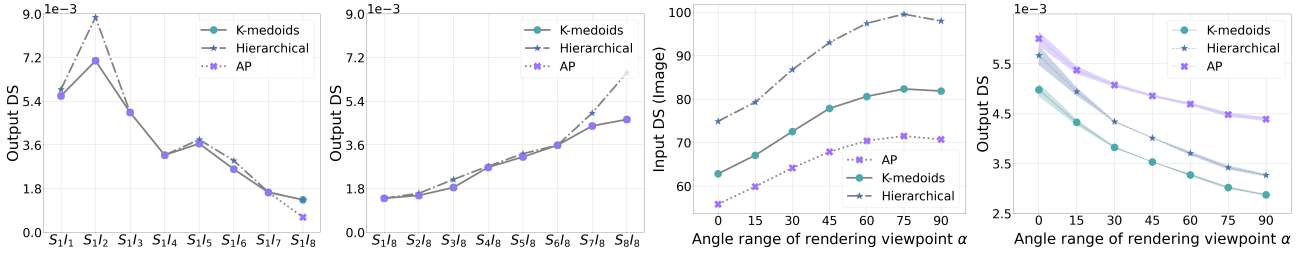


Figure 9: Visualization of input images and shapes in OC and VC coordinates of ShapeNet [3].



(a) (b) (c) (d)

Figure 10: DS from different clustering methods. (a)(b) Output DS of models trained on synthetic dataset with more dispersed training images (a) and shapes (b). (c)(d) Input/Ouput DS of models trained on ShapeNet with more dispersed training images.

C. Hyperparameter of Dispersion Score

In this section, we explain our method to tune the number of clusters (NC) in the proposed DS metric. To choose the best NC, we do a parameter sweeping and choose the value equal to the elbow of the DS curves. We automatically determine the elbow using the "Kneedle" method in [18]. This method approximately finds the point with the maximum curvature using a score called "normalized distance" and selects that as the elbow. See Figure 11. For each experiment, we compute the mean of the normalized distance across different trained models to determine the NC. For the

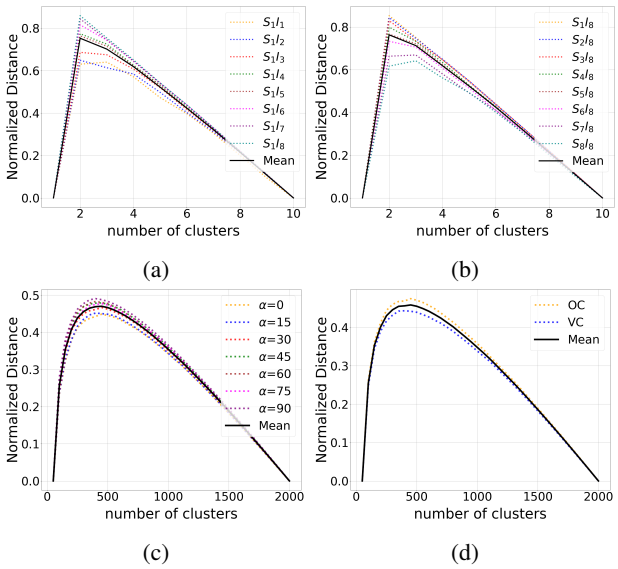


Figure 11: The normalized distance computed in Kneedle. (a)(b) Synthetic dataset with more dispersed training images and shapes. (c)(d) ShapeNet with more dispersed training images and shapes.

synthetic dataset, the NC is 2. For ShapeNet, the NC is 500.

D. Synthetic Data Generation

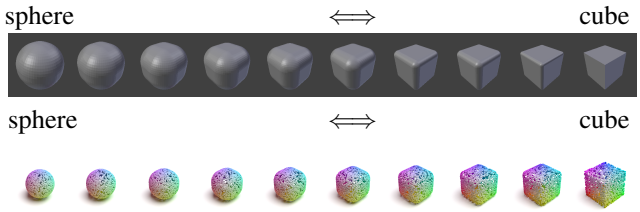


Figure 12: Image and shape examples of the synthetic dataset built by interpolating between a sphere and a cube. *Upper row* shows rendered images. *Lower row* shows shapes represented by point clouds.

We provide the details of synthetic data generation in this section. The image and shape examples are shown in Figure 12. We use the software Blender to generate base shapes in a mesh format. Then, we use the Shrinkwrap modifier in the "Nearest Vertex" mode to define the shape morphing between the two base shapes and control the interpolation progress by the Blender Shape Keys panel. After creating the mesh dataset, we render images and sample point clouds from meshes.

E. Implementation Details

In this section, we provide implementation details, including baseline models and training protocol.

Baselines For NN-based methods, we include PSGN [6], FoldingNet [28], AtlasNet-Sphere [9], AtlasNet-25 [9]. We use a ResNet-18 image encoder without any pre-training, the encoder outputs a 1024 dimensional latent vector. We use the same image encoder for all the models. We implement the decoder of each model according to architectures in the original publications.

Training Protocol Among all the experiments, the loss function is Chamfer distance [6], and optimizer is Adam [14]. For experiments on the synthetic dataset, each model is trained for 3600 iterations, using batch size 8. The initial learning rate is 1e-3, and the learning rate decays at 2400, 3000, 3300 iterations by a ratio of 0.1. For experiments on ShapeNet, each model is trained for 120 epochs, the batch size is 64, the initial learning rate is 1e-3, and it decays at 90, 110, 115 epoch by ratio 0.1. The weight decay is set to be 0.

F. Verifying the Dispersion Relationship on SDF-based NNs

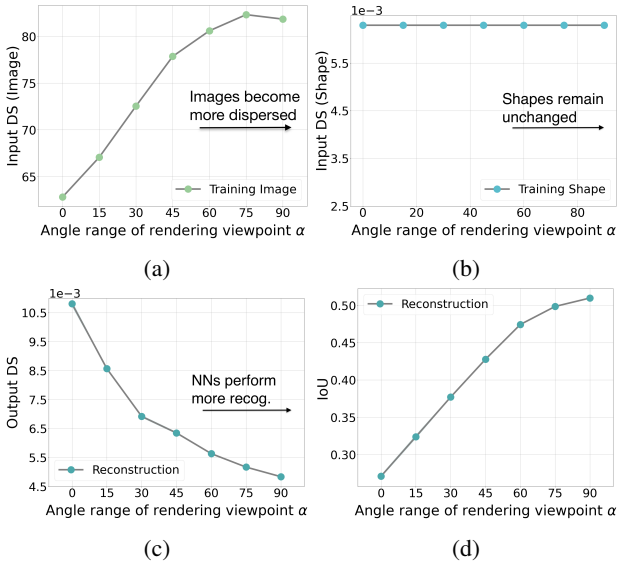


Figure 13: Measuring input/output DS and IoU in OC coordinate when the training images are rendered by increasing viewpoint angle range α. The unit of α is degree. (a) (b) From left to right, training images become more dispersed and training shapes remain unchanged. (c) Reconstructed shapes become less dispersed. (d) IoU of prediction become larger, the larger the better.

In the main paper, we have verified our hypothesis of dispersion relationship using point-cloud-based NN methods. In this section, we conduct experiments with methods based on signed distance fields (SDF). We adopt Occupancy Network (ONet) [16] as the baseline. For the transition *more*

dispersed training images, we reuse the group of increasingly dispersed training image datasets and use the training protocol in Section 5.3. While the training images become increasingly dispersed, the DS of training shapes remains unchanged because we use OC coordinate. The input DS of training images and shapes are shown in Figure 13a and 13b. For evaluation, we use Volumetric IoU [16] to measure reconstruction quality. To calculate DS, we extract mesh from predicted SDF following [16] and uniformly sample 2500 points from each mesh surface. The following procedure is the same as Section 3.3, and we omit the details here.

As our main claim predicts, the models tend more towards recognition as the decreasing output DS shows in Figure 13c. We also observe that ONet trained on more dispersed images achieves improved Volumetric IoU, as shown in Figure 13d. We notice that the results of SDF-based experiment are consistent with the observations of point-cloud-based experiment shown in Figure 5, which further verifies that more dispersed training images make NNs do more recognition.