3D Point Capsule Networks

Yongheng Zhao [†] [✣] [•] Tolga Birdal [†] [*] [•] Haowen Deng [†] [*] Federico Tombari [†]

[†] Technische Universität München, Germany [✣] University of Padova, Italy
[*] Siemens AG, München, Germany

Abstract

In this paper, we propose 3D point-capsule networks, an auto-encoder designed to process sparse 3D point clouds while preserving spatial arrangements of the input data. 3D capsule networks arise as a direct consequence of our unified formulation of the common 3D auto-encoders. The dynamic routing scheme [30] and the peculiar 2D latent space deployed by our capsule networks bring in improvements for several common point cloud-related tasks, such as object classification, object reconstruction and part segmentation as substantiated by our extensive evaluations. Moreover, it enables new applications such as part interpolation and replacement.

1. Introduction

Fueled by recent developments in robotics, autonomous driving and augmented/mixed reality, 3D sensing has become a major research trend in computer vision. Conversely to RGB cameras, the sensors used for 3D capture provide rich geometric structure, rather than high-fidelity appearance information. This is proved advantageous for those applications where color and texture are insufficient to accomplish the given task, such as reconstruction/detection of texture-less objects. Unlike the RGB camera case, 3D data come in a variety of forms: range maps, fused RGB-D sequences, meshes and point clouds, volumetric data. Thanks to their capability of representing a sparse 3D structure accurately while being agnostic to the sensing modality, point clouds have been a widespread choice for 3D processing.

The proliferation of deep learning has recently leaped into the 3D domain and architectures for consuming 3D points have been proposed either for volumetric [28] or sparse [26] 3D representations. These architectures overcame many challenges brought in by 3D data, such as order-invariance, complexity due to the added data dimension and local density variations. Unfortunately they often discard spatial arrangements in data, hence falling short of respecting the parts-to-whole relationship, which is critical to explain and describe 3D shapes; maybe even more severe than in the 2D domain due to the increased dimensionality [2].

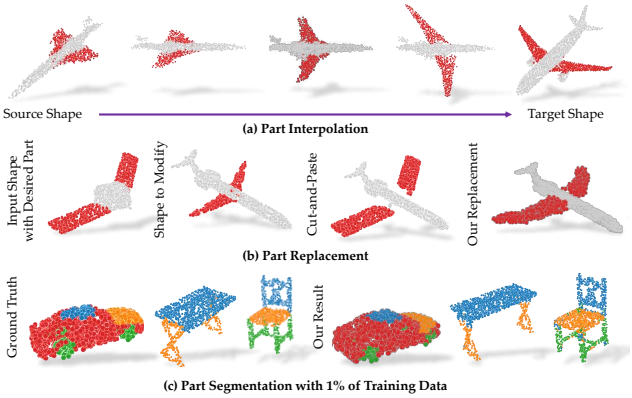


Figure 1. Our *3D-PointCapsNet* improves numerous 3D tasks while enabling interesting applications such as latent space part interpolation or complete part modification, an application where a simple cut-and-paste results in inconsistent outputs.

In this work we first present a unified look to some well known 3D point decoders. Within this view, and based on the renowned 2D capsule networks (CN) [30], we propose the unsupervised *3D point-capsule networks* (3D-PointCapsNet), an auto-encoder for generic representation learning in unstructured 3D data. Powered by the built-in routing-by-agreement algorithm [30], our network respects the geometric relationships between the parts, showing better learning ability and generalization properties. We design our 3D-PointCapsNet architecture to take into account the sparsity of point clouds by employing PointNet-like input layers [26]. Through an unsupervised dynamic routing, we organize the outcome of multiple max-pooled feature maps into a powerful latent representation. This intermediary latent space is parameterized by *latent capsules* - stacked latent activation vectors specifying the features of the shapes and their likelihood.

Latent capsules obtained from point clouds alleviate the restriction of parameterizing the latent space by a single,

[•] *First two authors contributed equally to this work.*

low dimensional vector; instead they give explicit control on the basis functions that get composed into 3D shapes. We further propose a novel 3D point-set decoder operating on these capsules, leading to better reconstructions with increased operational capabilities as shown in Fig. 1. These new abilities stem from the latent capsules instantiating as various shape parameters and concentrating not spatially but semantically across the shape under consideration, even when trained in an unsupervised fashion. We also propose to supply a limited amount of task-specific supervision such that the individual capsules can excel at solving individual sub-problems, e.g. if the task is part-based segmentation, they specialize on different meaningful parts of each shape.

Our extensive quantitative and qualitative evaluation demonstrates the superiority of our architecture. First, we advance the state of the art by a significant margin on multiple frontiers such as 3D local feature extraction, point cloud reconstruction and transfer learning. Next, we show that the distinct attention mechanism of the capsules, driven by dynamic routing, allows a wider range of 3D applications compared to the state of the art auto-encoders: a) part replacement, b) part-by-part animation via interpolation. Note that both of these tasks are non-trivial for standard architectures that rely on 1D latent vectors. Finally, we present improved generalization to unseen data, reaching accuracy levels up to 85% even when using 1% of training data. In a nutshell, our core contributions are:

1. Motivated by a unified perspective of the common point cloud auto-encoders, we propose capsule networks for the realm of 3D data processing as a powerful and effective tool.
2. We show that out point-capsule AE can surpass the current art in reconstruction quality, local 3D feature extraction and transfer learning for 3D object recognition.
3. We adapt our latent capsules to different tasks with semi-supervision and show that the latent capsules can master on peculiar parts or properties of the shape. In the end, this paves the way to higher quality predictions and a diverse set of applications like part specific interpolation.

Our source code is publicly available under:
`https://tinyurl.com/yxq2tmv3`.

2. Related Work

Point Clouds in Deep Networks Thanks to their generic capability of efficiently explaining 3D data without making assumptions on the modality, point clouds are the preferred containers for many 3D applications [48, 25]. Due to this widespread use, recent works such as PointNet [26], PointNet++ [27], SO-Net [22], spherical convolutions [20], Monte Carlo convolutions [12] and dynamic graph networks [44] have all devised point cloud-specific architectures that exploited the sparsity and permutation-invariant properties of 3D point sets. It is also common to process point sets by using local projections reducing the convolution operation down to two dimensions [34, 15].

Recently, unsupervised architectures followed up on their supervised counterparts. PU-Net [43] proposed better upsampling schemes to be used in decoding. FoldingNet [41] introduced the idea of deforming a 2D grid to decode a 3D surface as a point set. PPF-FoldNet [7] improved upon the supervised PPFNet [8] in local feature extraction by benefiting from FoldingNet's decoder [41]. AtlasNet [11] can be seen as an extension of FoldingNet to multiple grid patches and provided extended capabilities in data representation. PointGrow [32] devised an auto-regressive model for both unconditional and conditional point cloud generation leading to effective unsupervised feature learning. Achlioptas *et al*. [1] adapted GANs to 3D point sets, paving the way to enhanced generative learning.

2D Capsule Networks Thanks to their general applicability, capsule networks (CNs) have found tremendous use in 2D deep learning. LaLonde and Bagci [19] developed a deconvolutional capsule network, called *SegCaps*, tackling object segmentation. Durate *et al*. [9] extended CNs to action segmentation and classification by introducing *capsule-pooling*. Jaiswal *et al*. [16], Saqur *et al*. [31] and Upadhyay *et al*. [35] proposed Capsule-GANs, *i.e*. capsule network variants of the standard generative adversarial networks (GAN) [10]. These have shown better 2D image generation performance. Lin *et al*. [23] showed that capsule representations learn more meaningful 2D manifold embeddings than neurons in a standard CNN do.

There have also been significant improvements upon the initial CN proposal. Hinton *et al*. improved the routing by EM algorithm [13]. Wang and Liu saw the routing as an optimization minimizing a combination of clustering-like loss and a KL regularization term [36]. Chen and Crandall [6] suggested *trainable routing* for better clustering of capsules. Zhang *et al*. [47] unified the existing routing methods under one umbrella and proposed weighted kernel density estimation based routing methods. Zhang *et al*. [46] chose to use the norm to explain the existence of an entity and proposed to learn a group of capsule subspaces onto which an input feature vector is projected. Lenssen *et al*. [21] introduced guaranteed equivariance and invariance properties to capsule networks by the use of group convolutions.

3D Capsule Networks Up until now, the use of the capsule idea in the 3D domain has been a rather uncharted territory. Weiler *et al*. [38] rigorously formalized the convolutional capsules and presented a convolutional neural network (CNN) equivariant to rigid motions. Jimenez *et al*. [17] as well as Mobniy and Nguyen [24] extended capsules to deal with volumetric medical data. VideoCapsuleNet [9] also used a volumetric representation to handle

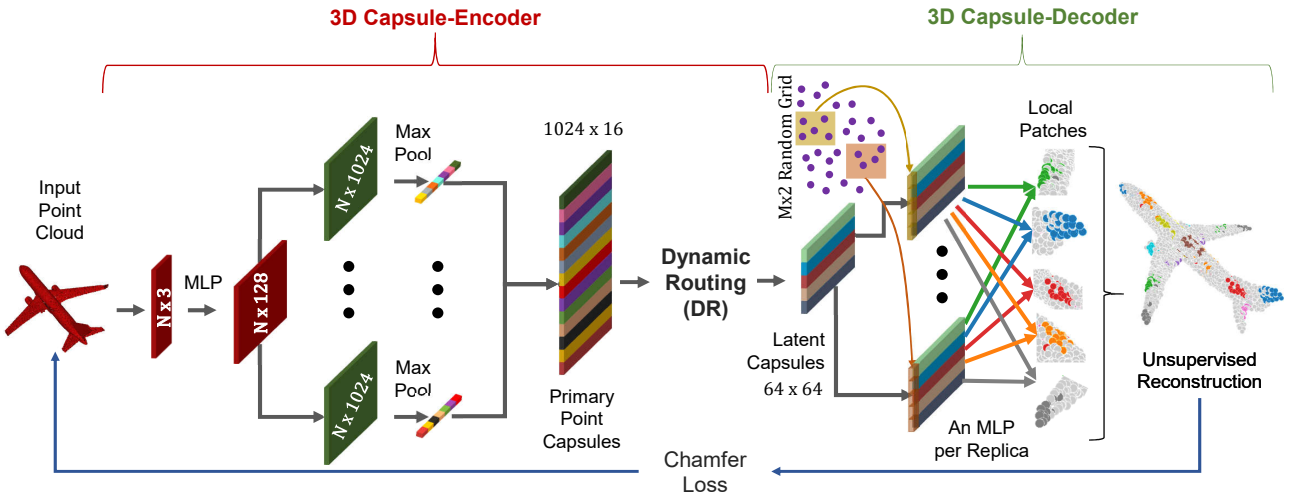


Figure 2. 3D Point Capsule Networks. Our capsule-encoder accepts an $N \times 3$ point cloud as input and uses an MLP to extract $N \times 128$ features from it. These features are then sent into multiple independent convolutional-layers with different weights, each of which is max-pooled to a size of 1024. The pooled features are then concatenated to form the *primary point capsules* (PPC) (1024×16). A subsequent dynamic routing clusters the PPC into the final *latent capsules*. Our decoder, responsible for reconstructing point sets given the latent features, endows the latent capsules with random 2D grids and applies MLPs $(64 - 64 - 32 - 16 - 3)$ to generate multiple point patches. These point patches target different regions of the shape thanks to the DR [30]. Finally, we collect all the patches into a final point cloud and measure the Chamfer distance to the input to guide the network to find the optimal reconstruction. In figure, part-colors encode capsules.

temporal frames of the video. Yet, to the best of our knowledge, we are the first to devise a capsule network specifically for 3D point clouds, exploiting their sparse and unstructured nature for representing 3D surfaces.

3. Method

3.1. Formulation

We first follow the AtlasNet convention [11] and present a unified view of some of the common 3D auto-encoders. Then, we explain our *3D-PointCapsNet* within this geometric perspective and justify its superiority compared to its ancestors. We will start by recalling the basic concepts:

Definition 1 (Surface and Point Cloud)
A 3D surface (shape) is a differentiable 2-manifold embedded in the ambient 3D Euclidean space: $\mathcal{M}^2 \in \mathbb{R}^3$. *We approximate a* ***point cloud*** *as a sampled discrete subset of the surface* $\mathbf{X} = \{\mathbf{x}_i \in \mathcal{M}^2 \cap \mathbb{R}^3\}$.

Definition 2 (Diffeomorphism)
A diffeomorphism is a continuous, invertible, structure-preserving map between two differentiable surfaces.

Definition 3 (Chart and Parametrization)
We admit an open set $U \in \mathbb{R}^2$ *and a diffeomorphism* $C : \mathcal{M}^2 \mapsto U \in \mathbb{R}^2$ *mapping an open neighborhood in 3D to its 2D embedding.* C *is called a* ***chart****. Its inverse,* $\Psi \equiv C^{-1} : \mathbb{R}^2 \mapsto \mathcal{M}^2$ *is called a* ***parameterization****.*

Definition 4 (Atlas)
A set of charts with images covering the 2-manifold is called an ***atlas****:* $\mathcal{A} = \cup_i C_i(\mathbf{x}_i)$.

A 3D auto-encoder learns to generate a 3D surface $\mathbf{X} \in \mathcal{M}^2 \cap \mathbb{R}^{N\times 3}$. By virtue of Dfn. 3 Ψ deforms a 2D point set to a surface. The goal of the generative models that are of interest here is to learn Ψ to best reconstruct $\hat{\mathbf{X}} \approx \mathbf{X}$:

Definition 5 (Problem)
Learning to generate the 2-manifolds is defined as finding function(s) $\Psi(U \,|\, \boldsymbol{\theta}) : \Psi(U \,|\, \boldsymbol{\theta}) \approx \mathbf{X}$ *[11].* $\boldsymbol{\theta}$ *is a lower dimensional parameterization of these functions:* $|\boldsymbol{\theta}| < |\mathbf{X}|$.

Theorem 1
Given that C^{-1} *exists,* Ψ*, chosen to be a 3-layer MLP, can reconstruct arbitrary 3D surfaces.*

Sketch of the proof. The proof is given in [41] and follows from the universal approximation theorem (UAT). □

Theorem 2
There exists an integer K *s.t. an MLP with* K *hidden units universally reconstruct* $\mathbf{X}$ *up to a precision* ϵ.

Sketch of the proof. The proof follows trivially from Thm. 1 and UAT [11]. □

Given these definitions, some of the typical 3D point decoders differentiate by making four choices [26, 11, 41]:

1. An open set U or discrete grid $\mathbf{U} \equiv \mathbf{P} = \{\mathbf{p}_i \in \mathbb{R}^2\}$.
2. Distance function $d(\mathbf{X}, \hat{\mathbf{X}})$ between the reconstruction $\hat{\mathbf{X}}$ and the input shape $\mathbf{X}$.
3. Parameterization function(s) Ψ.
4. Parameters $(\boldsymbol{\theta})$ of Ψ: $\Psi(U \,|\, \boldsymbol{\theta})$.

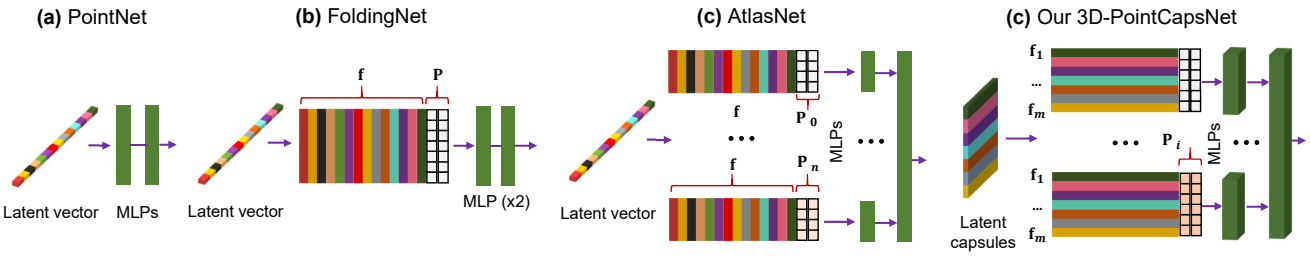


Figure 3. Comparison of four different state-of-the-art 3D point decoders. PointNet uses a single latent vector, and no surface assumption. Thus, $\boldsymbol{\theta}_{\text{pointnet}} = \mathbf{f}$. FoldingNet [41] learns a 1D latent vector along with a fixed 2D grid $\boldsymbol{\theta}_{\text{folding}} = \{\mathbf{f}, \mathbf{P}\}$. The advanced AtlasNet [11] learns to deform multiple 2D configurations onto local 2-manifolds: $\boldsymbol{\theta}_{\text{atlas}} = \{\mathbf{f}, \{\mathbf{P}_i\}\}$. Our point-capsule-network is capable of learning multiple latent representations each of which can fold a distinct 2D grid onto a specific local patch, $\boldsymbol{\theta}_{\text{ours}} = \{\{\mathbf{f}_i\}, \{\mathbf{P}_i\}\}$

One of the first works in this field, PointNet [26] is extended naturally to an AE by [1] making arguably the simplest choice. We will refer to this variant as *PointNet*. It lacks the grid structure $U = \varnothing$ and functions Ψ only depend upon a single latent feature: $\Psi(U \,|\, \boldsymbol{\theta}) = \Psi(\boldsymbol{\theta}) = \text{MLP}(\cdot \,|\, \mathbf{f} \in \mathbb{R}^k)$. FoldingNet uses a two-stage MLP as Ψ to warp a fixed grid $\mathbf{P}$ onto $\mathbf{X}$. A transition from FoldingNet to AtlasNet requires having multiple MLP networks operating on multiple 2D sets $\{\mathbf{P}_i\}$ constructed randomly on the domain $]0,1[^2$: $\boldsymbol{\mathcal{U}}(0,1)$. These explain the better learning capacity of AtlasNet: different MLPs learn to reconstruct distinct local surface patches by learning different charts.

Unfortunately, while numerous charts can be defined in the case of AtlasNet, all of the methods above still rely on a single latent feature vector, replicated and concatenated with U to create the input to the decoders. However, point clouds are found to consist of multiple basis functions [33] and having a single representation governing them all is not optimal. We opt to go beyond this restriction and choose to have a set of latent features $\{\mathbf{f}_i\}$ to capture different, meaningful basis functions.

With the aforementioned observations we can now re-write the well known 3D auto-encoders and introduce a new decoder formulation:

PointNet [26]	AtlasNet [11]	
$\mathbf{U} = \mathbf{P} = \varnothing$	$\mathbf{U} = \{\mathbf{P}_i\} : \mathbf{P}_i \in \boldsymbol{\mathcal{U}}(0,1)$	(1)
$\Psi(\boldsymbol{\theta}) := \text{MLP}(\cdot)$	$\Psi(\boldsymbol{\theta}) := \{\text{MLP}_i(\cdot)\}$	(2)
$\boldsymbol{\theta} := \mathbf{f}$	$\boldsymbol{\theta} := \{\mathbf{f}, \{\mathbf{P}_i\}\}$	(3)
$d(\mathbf{X}, \hat{\mathbf{X}}) := d_{\text{EMD}}(\mathbf{X}, \hat{\mathbf{X}})$	$d(\mathbf{X}, \hat{\mathbf{X}}) := d_{\text{CH}}(\mathbf{X}, \hat{\mathbf{X}})$	(4)
FoldingNet [41]	**Ours**	
$\mathbf{U} = \mathbf{P} = \mathbf{G}^{M \times M}$	$\mathbf{U} = \{\mathbf{P}_i\} : \mathbf{P}_i \in \boldsymbol{\mathcal{U}}(0,1)$	(5)
$\Psi(\boldsymbol{\theta}) := \text{MLP}(\text{MLP}(\cdot))$	$\Psi(\boldsymbol{\theta}) := \{\text{MLP}_i(\cdot)\}$	(6)
$\boldsymbol{\theta} := \{\mathbf{f}, \mathbf{P}\}$	$\boldsymbol{\theta} := \{\mathbf{F} \triangleq \{\mathbf{f}_i\}, \{\mathbf{P}_i\}\}$	(7)
$d(\mathbf{X}, \hat{\mathbf{X}}) := d_{\text{CH}}(\mathbf{X}, \hat{\mathbf{X}})$	$d(\mathbf{X}, \hat{\mathbf{X}}) := d_{\text{CH}}(\mathbf{X}, \hat{\mathbf{X}})$	(8)

where d_{EMD} is the Earth Mover [29] and d_{CH} is the Chamfer distance. $\mathbf{G}^{M \times M} = \{(i \otimes j) : \forall i, j \in [0, \ldots, \frac{M-1}{M}]\}$ is a 2D uniform grid. $\mathbf{f} \in \mathbb{R}^k$ represents a k-dimensional latent vector. $\boldsymbol{\mathcal{U}}(a,b)$ depicts an open set defined by a uniform random distribution in the interval $]a,b[^2$.

Note that it is possible to easily mix these choices to create variations[‡]. Though, many interesting architectures only optimize for a single latent feature $\mathbf{f}$. To the best of our knowledge, one promising direction is taken by the capsule networks [14], where multitudes of convolutional filters enable the learning of a collection of *capsules* $\{\mathbf{f}_i\}$ thanks to the dynamic routing [30]. Hence, we learn our parameters $\{\boldsymbol{\theta}_i\}$ by devising a new point cloud *capsule decoder* that we coin *3D-PointCapsNet*. We illustrate the choices made by four AEs under this unifying umbrella in Fig. 3.

3.2. 3D-PointCapsNet Architecture

We now describe the architecture of the proposed *3D-PointCapsNet* as a deep 3D point cloud auto-encoder, whose structure is depicted in Fig. 2.

Encoder The Input to our network is an $N \times d$ point cloud, where we fix $N = 2048$ and for typical point sets $d = 3$. Similarly to PointNet [26], we use a point-wise Multi-Layer Perceptron (MLP) $(3-64-128-1024)$ to extract individual local feature maps. In order to diversify the learning as suggested by capsule networks, we feed these feature maps into multiple independent convolutional layers with different weights, each with a distinct summary of the input shape with diversified attention. We then max-pool their responses to obtain a global latent representation. These descriptors are then concatenated into a set of vectors named *primary point capsules*, $\mathbf{F}$. Size of $\mathbf{F}$ depends upon the size $S_c := 1024$ and the number $K := 16$ of independent kernels at the last layer of MLP. We then use the dynamic routing [30] to embed the primary point capsules into higher level *latent capsules*. Each capsule is independent and can be considered as a *cluster centroid* (codeword) of the primary point capsules. The total size of the latent capsules is fixed to 64×64 (*i.e.*, 64 vectors each sized 64).

[‡]FoldingNet presents evaluations with random grids in their appendix.

Table 1. Descriptor matching results (recall) on the standard 3DMatch benchmark [45, 7].

	Kitchen	Home 1	Home 2	Hotel 1	Hotel 2	Hotel 3	Study	MIT Lab	Average
3DMatch [45]	0.5751	0.7372	0.7067	0.5708	0.4423	0.6296	0.5616	0.5455	0.5961
CGF [18]	0.4605	0.6154	0.5625	0.4469	0.3846	0.5926	0.4075	0.3506	0.4776
PPFNet [8]	0.8972	0.5577	0.5913	0.5796	0.5769	0.6111	0.5342	0.6364	0.6231
FoldNet [41]	0.5949	0.7179	0.6058	0.6549	0.4231	0.6111	0.7123	0.5844	0.6130
PPF-FoldNet-2K [7]	0.7352	0.7564	0.625	0.6593	0.6058	0.8889	0.5753	0.5974	0.6804
PPF-FoldNet-5K [7]	0.7866	0.7628	0.6154	0.6814	0.7115	**0.9444**	0.6199	0.6234	0.7182
Ours-2K	**0.8518**	**0.8333**	**0.7740**	**0.7699**	**0.7308**	**0.9444**	**0.7397**	**0.6494**	**0.7867**

Decoder Our decoder treats the latent capsules as a feature map and uses MLP$(64-64-32-16-3)$ to reconstruct a patch of points $\hat{\mathbf{X}}_i$, where $|\hat{\mathbf{X}}_i| = 64$. At this point, instead of replicating a single vector as done in [41, 11], we replicate the entire capsule m times and to each replica we append a unique randomly synthesized grid $\mathbf{P}_i$ specializing it to a local area. This further stimulates the diversity. We arrive at the final shape $\hat{\mathbf{X}}_i$ by propagating the replicas through a final MLP for each patch and gluing the output patches together. We choose $m = 32$ to reconstruct $|\hat{\mathbf{X}}| = 32 \times 64 = 2048$ points, the same amount as the input. Similar to other AEs, we approximate the loss over 2-manifolds by the discrete Chamfer metric:

$$d_{CH}(\mathbf{X}, \hat{\mathbf{X}}) = \frac{1}{|\mathbf{X}|} \sum_{\mathbf{x} \in \mathbf{X}} \min_{\hat{\mathbf{x}} \in \hat{\mathbf{X}}} \|\mathbf{x} - \hat{\mathbf{x}}\|_2 + \frac{1}{|\hat{\mathbf{X}}|} \sum_{\hat{\mathbf{x}} \in \hat{\mathbf{X}}} \min_{\mathbf{x} \in \mathbf{X}} \|\mathbf{x} - \hat{\mathbf{x}}\|_2 \quad (9)$$

However, this time $\hat{\mathbf{X}}$ follows from the capsules: $\hat{\mathbf{X}} = \cup_i \Psi_i(\mathbf{P}_i | \{\mathbf{f}_i\})$.

Incorporating Optional Supervision Motivated by the regularity of capsule distribution over the 2-manifold, we created a capsule-part network that spatially segments the object by associating capsules to parts. The goal here is to assign each capsule to a single part of the object. Hence, we treat this part-segmentation task as a per-capsule classification problem, rather than a per-point one as done in various preceding algorithms [26, 27]. This is only possible due to the spatial attention of the capsule networks.

The input of capsule-part network is the latent-capsules obtained from the pre-trained encoder. The output is the part label for each capsule. The ground truth (GT) capsule labeling is obtained from the ShapeNet-Part dataset [42] in three steps: 1) reconstructing the local part given the capsule and a pre-trained decoder, 2) retrieving the label of the nearest neighbor (NN) GT point for each reconstructed point, 3) computing the most frequent one (mode) among the retrieved labels.

To associate a part to a capsule, we use a shared MLP with a cross entropy loss to classify the latent capsules into parts. This network is trained independently from the 3D-PointCapsNet AE for part supervision. We provide additional architectural details in the supplementary material.

4. Experiments

We evaluate our method first quantitatively and then qualitatively on numerous challenging 3D tasks such as local feature extraction, point cloud classification, reconstruction, part segmentation and shape interpolation. We also include a more specific application of *latent space part-interpolation* that is made possible by the use of capsules. For evaluation regarding these tasks, we use multiple benchmark datasets: ShapeNet-Core [5], Shapenet-Part [42], ModelNet40 [40] and 3DMatch benchmark [45].

Implementation Details Prior to training, the input point clouds are aligned to a common reference frame and size normalized. To train our network we use an ADAM optimizer with an initial learning rate of 0.0001 and a batch size of 8. We also employ batch normalization (BN) and RELU activation units at the point of feature extraction to generate primary capsules. Similarly, the multi-stage MLP of the decoder also uses a BN and RELU units except for the last layer, where the activations are scaled by a $tanh(\cdot)$. During dynamic routing operation, we use the squash activation function mentioned in [30, 14].

4.1. Quantitative Evaluations

3D Local Feature Extraction We first evaluate 3D PointCapsule Networks on the challenging task of local feature extraction from point cloud data. In this domain, learning methods have already outperformed their handcrafted counterparts by a large margin and hence, we compare only against those, namely 3DMatch [45], PPFNet [8], CGF [18] and PPF-FoldNet [7]. PPF-FoldNet is completely unsupervised and yet is still the top performer, thanks to the FoldingNet [41] encoder-decoder. It is thus intriguing to see how its performance is affected if one simply replaces its FoldingNet auto-encoder with 3D-PointCapsNet. In an identical setting as [7], we learn to reconstruct the 4 dimensional point pair features [3, 4] of a local patch, instead of the 3D

Table 2. Descriptor matching results (recall) on the rotated 3DMatch benchmark [45, 7].

	Kitchen	Home 1	Home 2	Hotel 1	Hotel 2	Hotel 3	Study	MIT Lab	Average
3DMatch [45]	0.0040	0.0128	0.0337	0.0044	0.0000	0.0096	0.0000	0.0260	0.0113
CGF [18]	0.4466	0.6667	0.5288	0.4425	0.4423	0.6296	0.4178	0.4156	0.4987
PPFNet [8]	0.0020	0.0000	0.0144	0.0044	0.0000	0.0000	0.0000	0.0000	0.0026
FoldNet [41]	0.0178	0.0321	0.0337	0.0133	0.0096	0.0370	0.0171	0.0260	0.0233
PPF-FoldNet-2K [7]	0.7352	0.7692	0.6202	0.6637	0.6058	0.9259	0.5616	0.6104	0.6865
PPF-FoldNet-5K [7]	0.7885	0.7821	0.6442	0.6770	0.6923	**0.9630**	0.6267	0.6753	0.7311
Ours-2K	**0.8498**	**0.8525**	**0.7692**	**0.8141**	**0.7596**	0.9259	**0.7602**	**0.7272**	**0.8074**

Table 3. Evaluating reconstruction quality. Oracle refers to a random sampling of the input 3D shape and constitutes an lower bound on what is achievable. The Chamfer Distance is multiplied by 10^3 for better viewing. CD denotes *Chamfer distance* and PB refers to *Point Baseline*.

	Oracle	PB	AtlasNet-25	AtlasNet-125	Ours
CD	0.85	1.91	1.56	1.51	**1.46**

space of points, and use the latent capsule (codeword) as a 3D descriptor. To restrict the feature vector to a reasonable size of 512, we limit ourselves only to 16×32 capsules. We then run the matching evaluation on the 3DMatch Benchmark dataset [45] as detailed in [7], and report the recall of correctly founded matches after 21 epochs in Tab. 1.

We note that our point-capsule networks exhibit an advanced capacity for learning local features, surpassing the state of the art by 10% on the average, even when using $2K$ points unlike the $5K$ of PPF-FoldNet. It is also noteworthy that, except for the Kitchen sequence where PPFNet shows remarkable performance, the recall attained by our network consistently remains above all others. We believe that such dramatic improvement is related to the robustness of capsules towards slight deformations in the input data, as well as to our effective decoder.

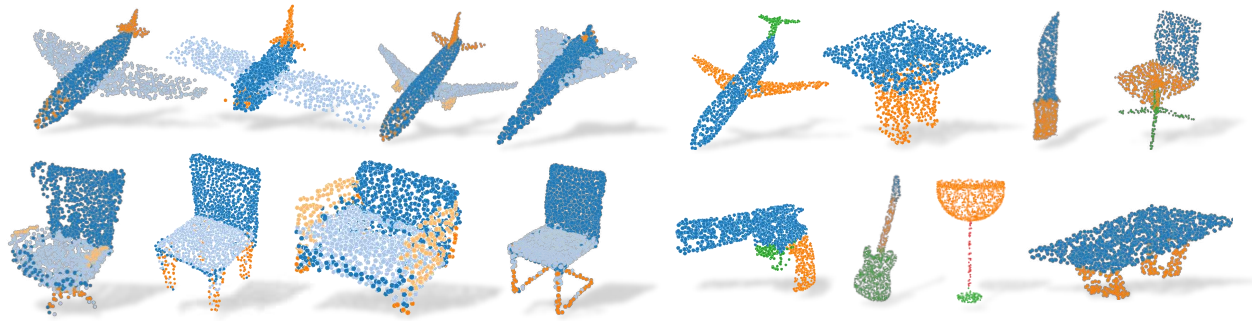
Do Our Features Also Perform Well Under Rotation? PPF local encoding of PPF-FoldNet is rotation-invariant. Being based on the same representation, our local feature network should enjoy similar properties. It is of interest to see whether the good performance attained on the standard 3DMatch benchmark transfers to more challenging scenes demanding rotation invariance. To this aim, we repeat the previous assessment on the Rotated-3DMatch benchmark [7], a dataset that introduces arbitrary rotations to the scenes of [45]. Since this dataset contains 6DoF scene transformations, many methods that lack theoretical invariance, e.g. 3DMatch, PPFNet and FoldingNet simply fail. Our unsupervised capsule AE, however, is once again the top performer, surpassing the state of the art by $\sim 12\%$ on $2K$-point case as shown in Tab. 2. This significant gain justifies that our encoder manages to operate also on the space of 4D PPFs, holding on the theoretical invariances.

Table 4. Accuracy of classification by transfer learning on the ModelNet40 dataset. Networks are trained out ShapeNet55, except *Ours-Parts* that is trained on smaller ShapeNet-Parts dataset.

	Latent-GAN[1]	FoldingNet[41]	Ours-Parts	Ours
Acc.	85.7	88.4	88.9	**89.3**

3D Reconstruction In a further experiment, we evaluate the quality of our architecture in point generation. We assess the reconstruction performance by the standard Chamfer metric and base our comparisons on the state of the art auto-encoder AtlasNet and its baselines (point-MLP) [11]. We rely on the ShapeNet Core v2 dataset [5], using the same training and test splits as well as the same evaluation metric as those in AtlasNet's [11]. We show in Tab. 3 the Chamfer distances averaged over all categories and for $N > 2K$ points. It is observed that our capsule AE results in lower reconstruction error even when a large number of patches (125) is used in favor of AtlasNet. This justifies that the proposed network has a better summarization capability and can result in higher fidelity reconstructions.

Transfer Learning for 3D Object Classification In this section, we demonstrate the efficiency of learned representation by evaluating the classification accuracy obtained by performing transfer learning. Identical to [39, 1, 41], we train a linear SVM classifier so as to regress the shape class given the latent features. To do that, we reshape our latent capsules into a one dimensional feature and train the classifier on Modelnet40 [40]. We use the same train/test split sets as [41] and obtain the latent capsules by training 3D-PointCapsNet on a different dataset, the ShapeNet-Parts [42]. The training data has 14,000 models subdivided into 16 classes. The evaluation result is shown in Tab. 4, where our AE, trained on a smaller dataset compared to the ShapeNet55 of [1, 41] is capable of performing at least on par or better. This shows that learned latent capsules can handle smaller datasets and generalize better to new tasks. We also evaluated our classification performance when the training data is scarce and obtained similar result as the FoldingNet, $\sim 85\%$ on $\sim 20\%$ of training data.



(a) Unprocessed part segmentation on same class

(b) Part segmentation of multiple objects of different class

Figure 4. Part segmentation by capsule association. Having pre-trained the auto-encoder, we append a final part-supervision layer and use a limited amount of data to specialize the capsules on object parts. (**a**) across the shapes of the same class capsules capture semantic regions. (**b**) inter-class part segmentation. Colors indicate different capsule groups and (**b**) uses only a simple median filter to smooth the results.

Table 5. Part segmentation on ShapeNet-Part by learning only on the $x\%$ of the training data.

Metric	SONet-1%	Ours-1%	SONet-5%	Ours-5%
Accuracy	0.78	**0.85**	0.84	**0.86**
IoU	0.64	**0.67**	0.69	**0.70**

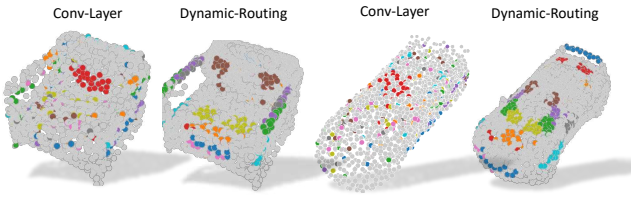


Figure 5. Distribution of 10 randomly selected capsules on the reconstructed shape after unsupervised autoencoder training **a**) with dynamic routing, **b**) with a simple convolutional layer.

4.2. Qualitative results

3D Object Part Segmentation with Limited Data We now demonstrate the regional attention of our latent capsule and their capacity to learn with limited data. To this end, we trained 3D-PointCapsNet on the ShapeNet-Part dataset [5] for part segmentation as explained in § 3, with a supervision of only $1-5\%$ part labeled training data. We tested our network on all of the available test data. To specialize the capsules to distinct parts, we select as many capsules as the part labels and let the per-capsule classification coincide to part predictions. Predicted capsule labels are propagated to the related points. For the sake of space, we compared our results only with the state of the art on this dataset, the SO-Net [23]. We use identical evaluation metrics as SO-Net [23]: *Accuracy* and *IoU* (Intersection over Union), and report our findings in Tab. 5. Note that, when trained on 1% of input data, we perform 7% better than SO-Net. When the amount of training data is increased to 5%, the gap closes but we still surpass SO-Net by 2%, albeit training a smaller network to classify latent-capsules rather than 3D points.

Does unsupervised training lead to specialized capsules? It is of interest to see whether the original argument of the capsule networks [30, 14] claiming to better capture the intrinsic geometric properties of the object still holds in the case of our unsupervised 3D-AE. To this aim, we first show in Fig. 5 that even with lack of supervision the capsules specialize on local parts of the model. While these parts may sometimes not correspond to the human annotated part segmentation of the model, we still expect them to concentrate on semantically similar regions of the 2-manifold. Fig. 5 visualizes the distribution of 10 capsules by coloring them individually and validates our argument.

To validate our second hypothesis, stating that such clustering arises thanks to the dynamic routing, we replace the DR part of the AE with standard PointNet-like layers projecting the 1024×64 PPC to 64^2 capsules and repeat the experiment. Fig. 5 depicts the spread of the latent vectors over the point set when such layer is employed as opposed to DR. Note that using this simple layer instead of DR both harms the reconstruction quality and yields an undesired spread of the capsules across the shape. We leave it as a future work to study the DR energy theoretically and provide more details on this experiment in the supplement.

Semi-supervision guides the capsules to meaningful parts. We now consider the effect of training in steering the capsules towards the optimal solution in the task of supervised part segmentation. First, we show in Fig. 4 the results obtained by the proposed part segmentation: (**a**) shows part segmentation across multiple shapes of the same class. These results are also unfiltered and the raw outcome of our network. (**b**) depicts part segmentation across a set of object classes from Shapenet-Part. It also shows that some minor confusions present in (**a**) can be corrected with a simple median filter. This is contrary and computationally preferable to costly CRFs smoothing the results [37].

Next, we observe that, as training iterations progress, the randomly initialized capsules specialize to parts, achieving

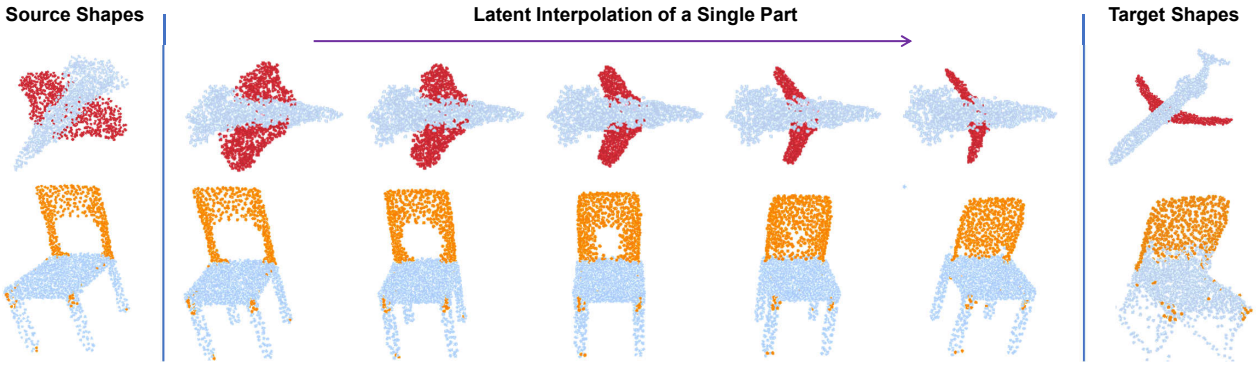


Figure 6. Part interpolation on the Shapenet-Part [42] dataset. (**left**) The source point cloud. (**right**) Target shape. (**middle**) Part interpolation. Fixed part is marked in light blue and the interpolated part is highlighted. Capsules are capable of performing part interpolation purely via latent space arithmetic.

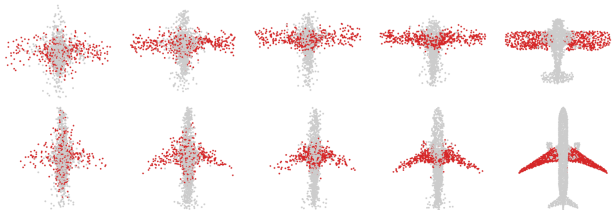


Figure 7. Visualizing the iterations of unsupervised AE training on the airplane object. For clear visualization, we fetch the colors belonging to the ∼20 capsules of the wing-part from our part predictions trained with part supervision.

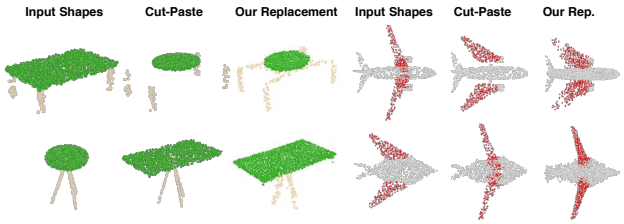


Figure 8. Part replacement. Performing replacement in the latent space rather than Euclidean space of 3D points yields geometrically consistent outcome.

a good part segmentation at the point of convergence. We visualize this phenomenon in Fig. 7, where the capsules that have captured the wings of the airplane are monitored throughout the optimization procedure. Even though the initial random distribution is spatially spread out, the resulting configuration is still part specific. This is a natural consequence of our capsule-wise part semi-supervision.

Part Interpolation / Replacement Finally, we explore the rather uncommon but particularly interesting application of interpolating, exchanging or switching object parts via latent-space manipulation. Thanks to the fact that 3D-PointCapsNet discovers multiple latent vectors specific to object attributes/shape parts, our network is capable of performing per-part processing in the latent space. To do that, we first spot a set of latent capsule pairs belonging to the same parts of two 3D point shapes and intersect them. Because these capsules explain the same part in multiple shapes, we assume that they are specific to the part under consideration and nothing else. We then interpolate linearly in the latent space between the selected capsules. As shown in Fig. 6 the reconstruction of intermediate shapes vary only at a single part, the one being interpolated. When the interpolator reaches the target shape it *replaces* the source part with the target one, enabling *part-replacement*. Fig. 8 further shows this in action. Given two shapes and latent capsules of the related parts, we perform a part exchange by simply switching some of the latent capsules and reconstructing. Conducting a part exchange directly on the input space by a cut-and-place would yield inconsistent shapes as the replaced parts would have no global coherence.

5. Conclusion

We have presented 3D Point-Capsule Network, a flexible and effective tool for 3D shape processing and understanding. We first presented a broad look to the common point cloud AEs. With the observation that a one dimensional latent embedding, the choice of the most preceding auto-encoders, is potentially sub-optimal, we opted to summarize the point clouds as a union of disjoint latent basis functions. We have shown that such choice can be implemented by learning the embedded *latent capsules* via dynamic routing. Our algorithm proved successful on an extensive evaluation on many 3D shape processing tasks such as 3D reconstruction, local feature extraction and part segmentation. Having a latent capsule set rather than a single vector also enabled us to address new applications such as part interpolation and replacement. In the future, we plan to deploy our network for pose estimation and object detection from 3D data, currently two of the key challenges in 3D computer vision.

Acknowledgements We would like to thank Yida Wang, Shuncheng Wu and David Joseph Tan for fruitful discussions. This work is partially supported by China Scholarship Council (CSC) and IAS-Lab in the Department of Information Engineering of the University of Padova, Italy.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 40–49. PMLR, 10–15 Jul 2018.

[2] Richard Bellman. *Dynamic programming*. Courier Corporation, 2013.

[3] Tolga Birdal and Slobodan Ilic. Point pair features based object detection and pose estimation revisited. In *2015 International Conference on 3D Vision*, pages 527–535. IEEE, 2015.

[4] Tolga Birdal and Slobodan Ilic. A point sampling algorithm for 3d matching of irregular geometries. In *2017 International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, 2017.

[5] 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.

[6] Zhenhua Chen and David Crandall. Generalized capsule networks with trainable routing procedure. *arXiv preprint arXiv:1808.08692*, 2018.

[7] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[8] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppfnet: Global context aware local features for robust 3d point matching. *Computer Vision and Pattern Recognition (CVPR). IEEE*, 1, 2018.

[9] Kevin Duarte, Yogesh Rawat, and Mubarak Shah. Videocapsulenet: A simplified network for action detection. In *Advances in Neural Information Processing Systems*, pages 7621–7630, 2018.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[11] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[12] Pedro Hermosilla, Tobias Ritschel, Pere-Pau Vázquez, Àlvar Vinacua, and Timo Ropinski. Monte carlo convolution for learning on non-uniformly sampled point clouds. In *SIGGRAPH Asia 2018 Technical Papers*, page 235. ACM, 2018.

[13] Geoffrey Hinton, Sara Sabour, and Nicholas Frosst. Matrix capsules with em routing. In *ICLR 2018 Conference Blind Submission*, 2018.

[14] Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders. In *International Conference on Artificial Neural Networks*, pages 44–51. Springer, 2011.

[15] Haibin Huang, Evangelos Kalogerakis, Siddhartha Chaudhuri, Duygu Ceylan, Vladimir G Kim, and Ersin Yumer. Learning local shape descriptors from part correspondences with multiview convolutional networks. *ACM Transactions on Graphics (TOG)*, 37(1):6, 2018.

[16] Ayush Jaiswal, Wael AbdAlmageed, Yue Wu, and Premkumar Natarajan. Capsulegan: Generative adversarial capsule network. In *European Conference on Computer Vision*, pages 526–535. Springer, 2018.

[17] Amelia Jiménez-Sánchez, Shadi Albarqouni, and Diana Mateus. Capsule networks against medical imaging data challenges. In *Intravascular Imaging and Computer Assisted Stenting and Large-Scale Annotation of Biomedical Data and Expert Label Synthesis*, pages 150–160. Springer, 2018.

[18] Marc Khoury, Qian-Yi Zhou, and Vladlen Koltun. Learning compact geometric features. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[19] Rodney LaLonde and Ulas Bagci. Capsules for object segmentation. *arXiv preprint arXiv:1804.04241*, 2018.

[20] Haun Lei, Naveed Akhtar, and Ajmal Mian. Spherical convolutional neural network for 3d point clouds. *arXiv preprint arXiv:1805.07872*, 2018.

[21] Jan Eric Lenssen, Matthias Fey, and Pascal Libuschewski. Group equivariant capsule networks. In *Advances in Neural Information Processing Systems*, pages 8858–8867, 2018.

[22] Jiaxin Li, Ben M Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9397–9406, 2018.

[23] Ancheng Lin, Jun Li, and Zhenyuan Ma. On learning and learned representation with dynamic routing in capsule networks. *arXiv preprint arXiv:1810.04041*, 2018.

[24] Aryan Mobiny and Hien Van Nguyen. Fast capsnet for lung cancer screening. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 741–749. Springer, 2018.

[25] Muzammal Naseer, Salman Khan, and Fatih Porikli. Indoor scene understanding in 2.5/3d for autonomous agents: A survey. *IEEE Access*, 7:1859–1887, 2019.

[26] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas 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*, pages 652–660, 2017.

[27] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017.

[28] Gernot Riegler, Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[29] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121, 2000.

[30] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, pages 3856–3866, 2017.

[31] Raeid Saqur and Sal Vivona. Capsgan: Using dynamic routing for generative adversarial networks. *arXiv preprint arXiv:1806.03968*, 2018.

[32] Yongbin Sun, Yue Wang, Ziwei Liu, Joshua E Siegel, and Sanjay E Sarma. Pointgrow: Autoregressively learned point cloud generation with self-attention. *arXiv preprint arXiv:1810.05591*, 2018.

[33] Minhyuk Sung, Hao Su, Ronald Yu, and Leonidas Guibas. Deep functional dictionaries: Learning consistent semantic structures on 3d models from functions. In *NIPS*, 2018.

[34] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3d. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3887–3896, 2018.

[35] Yash Upadhyay and Paul Schrater. Generative adversarial network architectures for image synthesis using capsule networks. *arXiv preprint arXiv:1806.03796*, 2018.

[36] Dilin Wang and Qiang Liu. An optimization view on dynamic routing between capsules. *ICLR Workshop Submission*, 2018.

[37] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):72, 2017.

[38] Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco Cohen. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. In *Advances in Neural Information Processing Systems*, pages 10402–10413, 2018.

[39] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016.

[40] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[41] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[42] Li Yi, Vladimir G Kim, Duygu Ceylan, I Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, Leonidas Guibas, et al. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (TOG)*, 2016.

[43] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2018.

[44] Ziwei Liu Sanjay E. Sarma Michael M. Bronstein Justin M. Solomon Yue Wang, Yongbin Sun. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018.

[45] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *CVPR*, 2017.

[46] Liheng Zhang, Marzieh Edraki, and Guo-Jun Qi. Cappronet: Deep feature learning via orthogonal projections onto capsule subspaces. In *Advances in Neural Information Processing Systems*, pages 5819–5828, 2018.

[47] Suofei Zhang, Quan Zhou, and Xiaofu Wu. *Fast Dynamic Routing Based on Weighted Kernel Density Estimation*. Springer International Publishing, 2020.

[48] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.