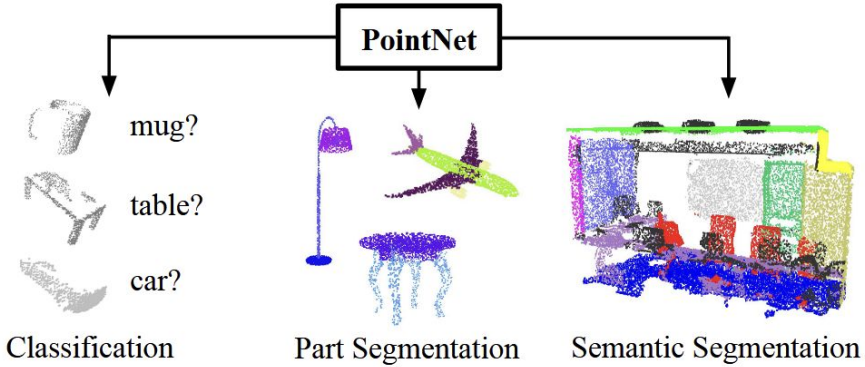
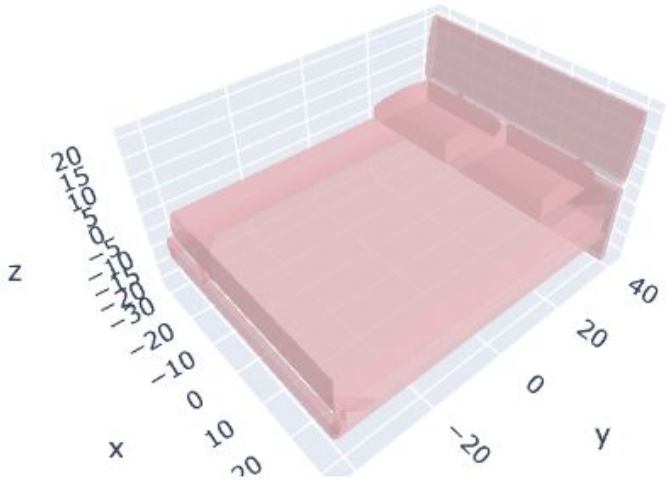
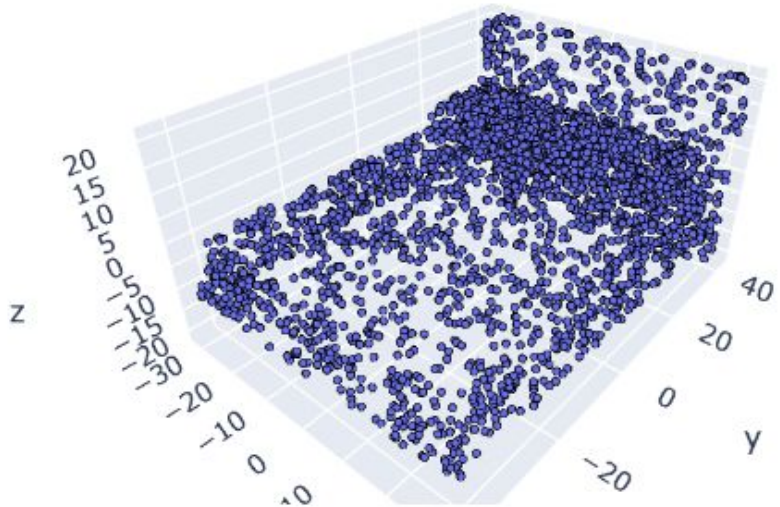
PointNet and PointNet++



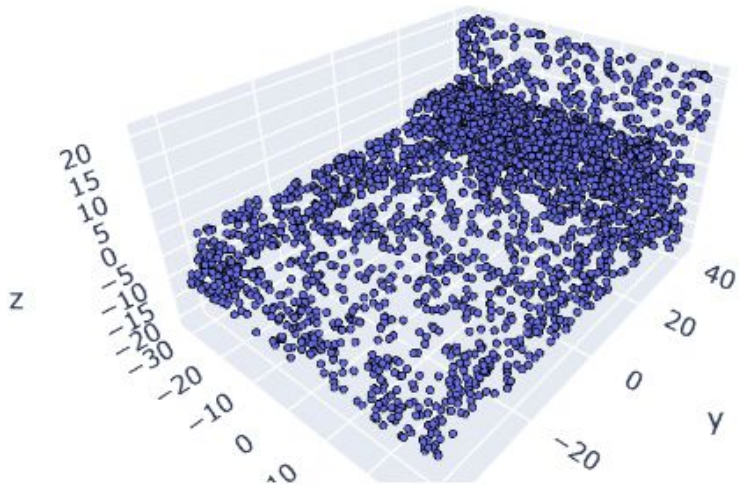
Paper Authors (2017): Charles R. Qi, Li Yi, Hao Su, Leonidas J. Guibas
Slides: Ana M. Cárdenas

Problem Statement



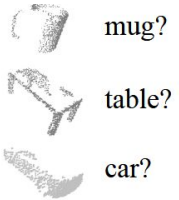


Problem Statement

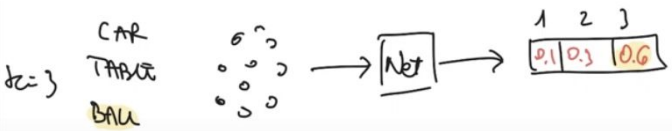


A point cloud (input) is represented as a set of 3D points $\{P_i \mid i = 1, \ldots, n\}$, where each point P_i is a vector of its (x, y, z) coordinate plus extra feature channels such as color, normal etc.

Problem Statement

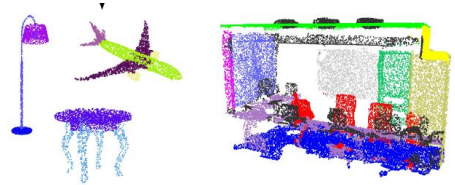


Classification

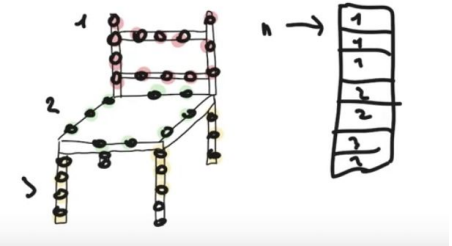


Object Classification

input: point cloud
output: outputs k scores for all the k candidate classes.



Part Segmentation Semantic Segmentation



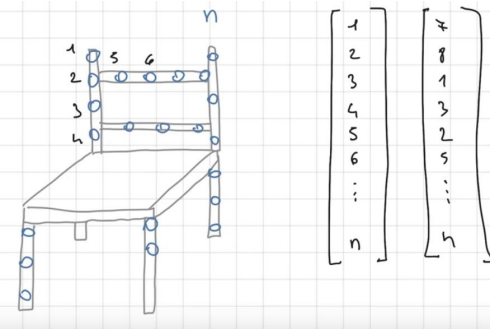
Segmentation

input: the input can be a single object for part region segmentation, or a sub-volume from a 3D scene for object region segmentation.
output: n × m scores for each of the n points and each of the m semantic sub- categories.

Properties of Point Sets in $\mathbb{R}^n$

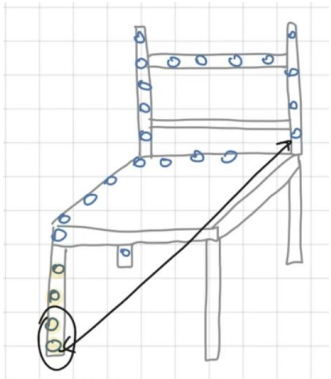
Unordered

Network should be Invariant to permutations of the input set of points



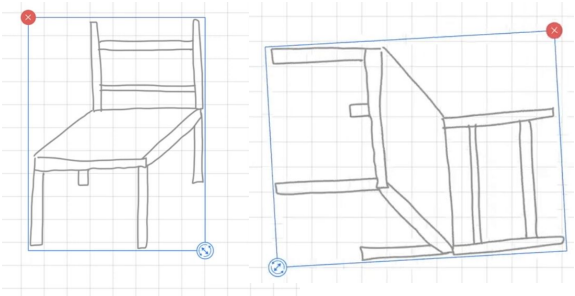
Interaction among points

Network should capture local structures from nearby points



Invariance under transformations

Network should be invariant to rotating and translating set of points all together



PointNet Architecture

Unordered

Symmetry Function for Unordered Input

Interaction among points

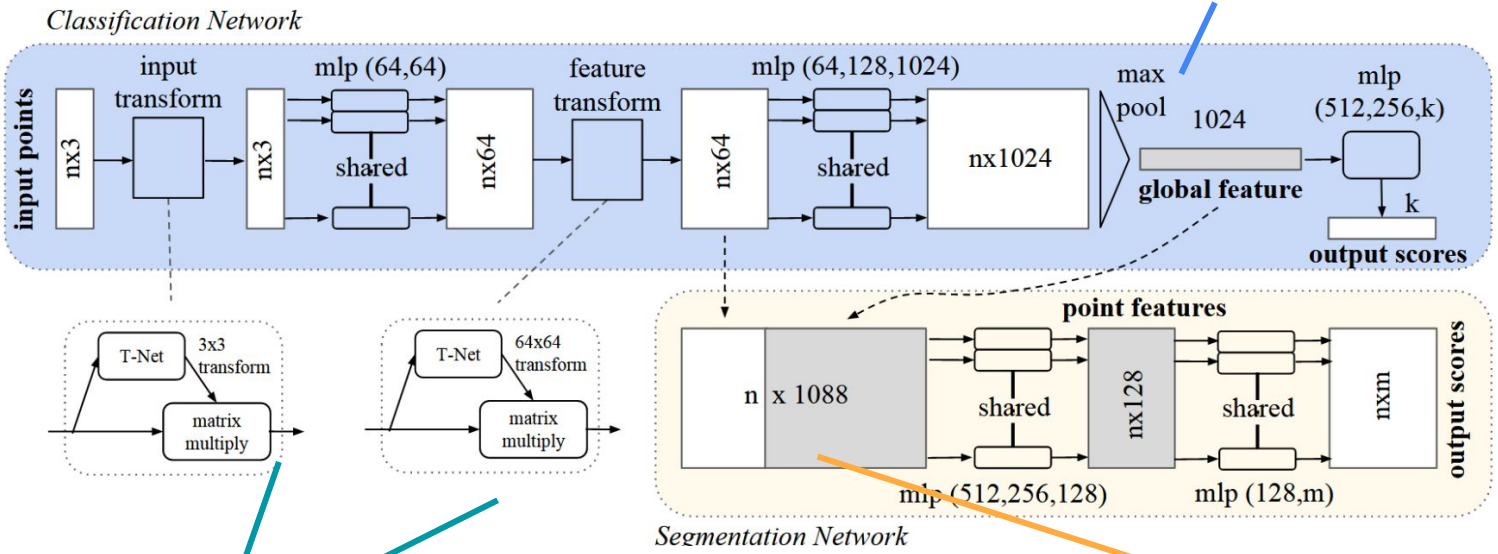
Local and Global Information Aggregation

Invariance under transformations

Joint Alignment Network

PointNet Architecture

Symmetry Function for Unordered Input



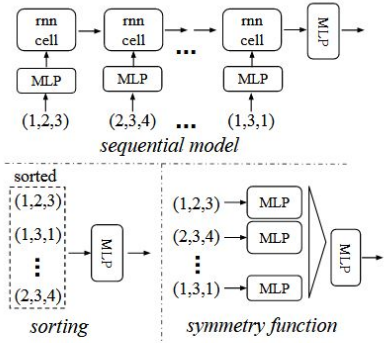
Joint Alignment Network

Local and Global Information Aggregation

Symmetry Function for Unordered Input

How to make a model invariant to input permutation?

- Sort input
- Use a RNN, but augment the training data by all kinds of permutations
- Use a symmetric function to aggregate the information from each point



	accuracy
MLP (unsorted input)	24.2
MLP (sorted input)	45.0
LSTM	78.5
Attention sum	83.0
Average pooling	83.8
Max pooling	**87.1**

Symmetry Function for Unordered Input

$$f(\{x_1, \ldots, x_n\}) \approx g(h(x_1), \ldots, h(x_n)),$$

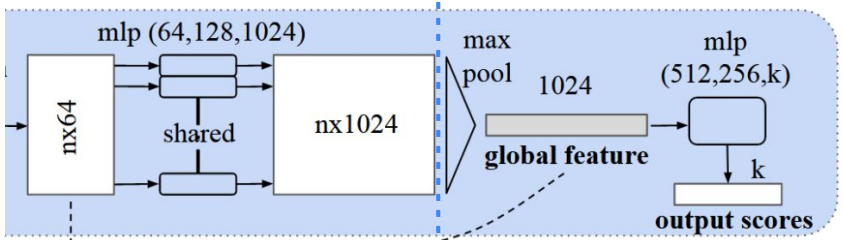
where $f : 2^{\mathbb{R}^N} \to \mathbb{R}, h : \mathbb{R}^N \to \mathbb{R}^K$ and $g : \underbrace{\mathbb{R}^K \times \cdots \times \mathbb{R}^K}_{} \to \mathbb{R}$ is a symmetric function.

Function g:

- Symmetric function (the order of its inputs does not affect the output) that aggregates the transformed points.
- Approximate g by a composition of a single variable function and a max pooling function

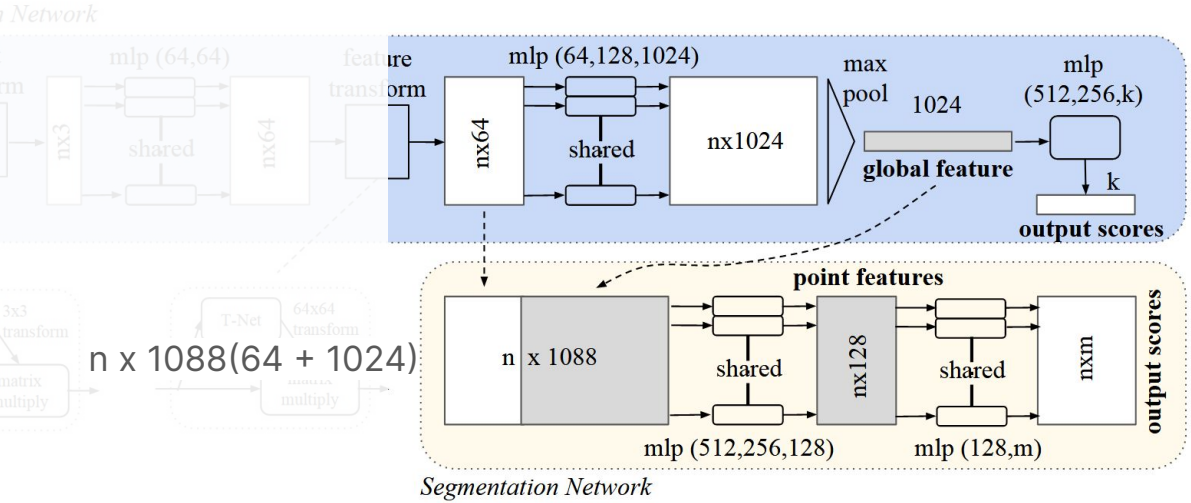
Function h:

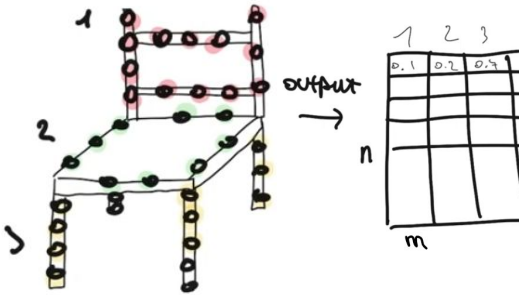
- transformation function that maps an individual point from its original space $\mathbb{R}^N$ to a new space $\mathbb{R}^K$.
- Approximate h by a multi-layer perceptron network



Local and Global Information Aggregation

- Point segmentation requires a combination of local and global knowledge
- They extract new per point features based on the combined point features and global features





Local and Global Information Aggregation

- The segmentation network can be trained to predict point normals, a local geometric property that is determined by a point's neighborhood.

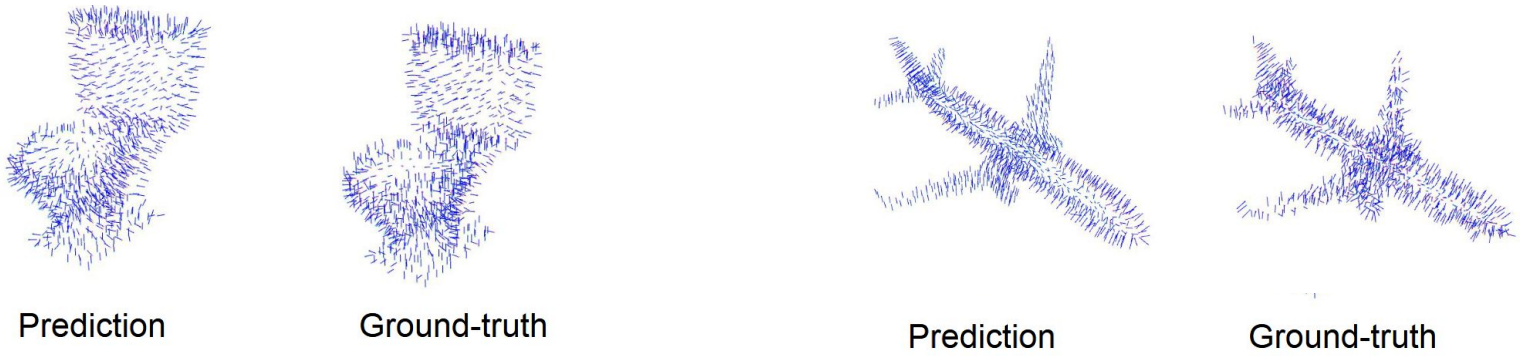
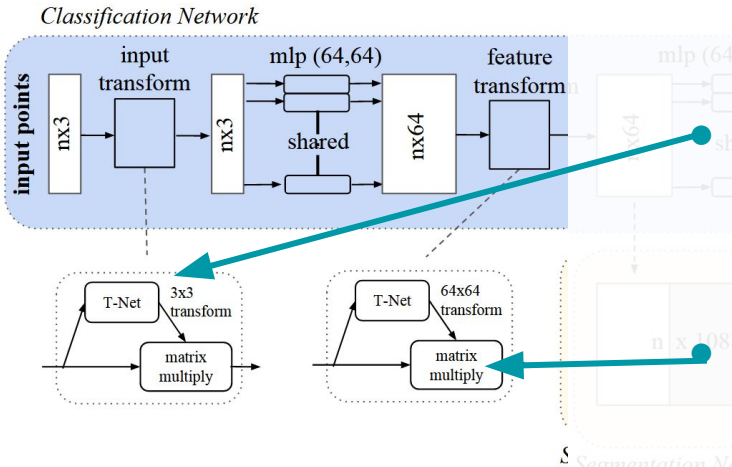


Figure 16. **PointNet normal reconstrution results.** In this figure, we show the reconstructed normals for all the points in some sample point clouds and the ground-truth normals computed on the mesh.

Joint Alignment Network



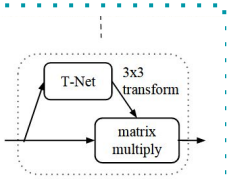
An alignment network on the input predicts a 3d transformation matrix to align all input set to a canonical space.

Another alignment network on point features and predict a feature transformation matrix to align features from different input point clouds.

Spatial transformer networks Jaderberg et al.

Joint Alignment Network

- The first transformation network is a “mini-PointNet”
  - Takes raw point cloud as input and regresses to a 3 × 3 matrix
  - The second transformation network has the same architecture except that the output is a 64 × 64 matrix.



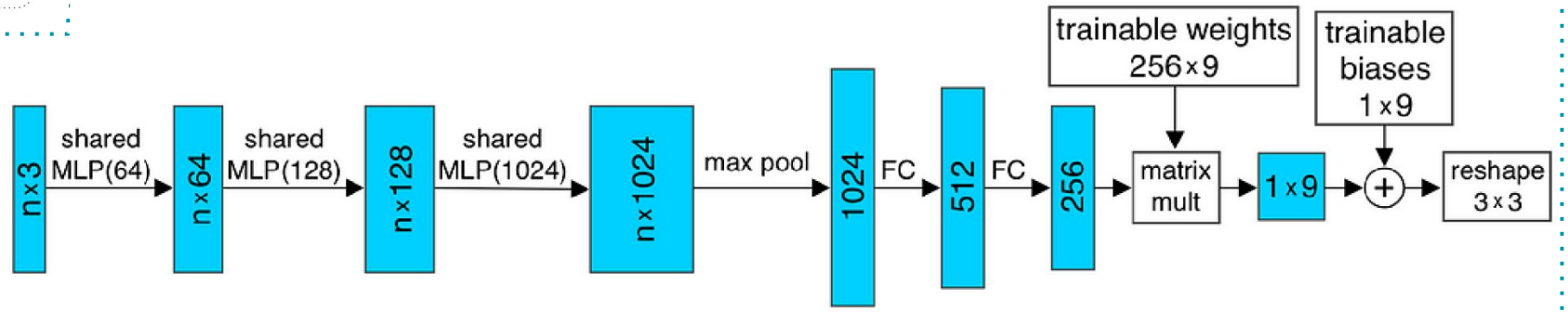


Fig. 8: Architecture of 3x3 T-Net

Joint Alignment Network

- The network enforces orthogonality of the transformation matrix with a regularization term

$$L_{reg} = \left\| I - AA^T \right\|_{F'}^2$$

Transform	accuracy
none	87.1
input (3x3)	87.9
feature (64x64)	86.9
feature (64x64) + reg.	87.4
both	**89.2**

Table 5. **Effects of input feature transforms.** Metric is overall classification accuracy on ModelNet40 test set.

Theoretical Analysis: Universal Approximation

- A small perturbation to the input point set should not greatly change the function values, such as classification or segmentation scores.
- In the worst case the network can learn to convert a point cloud into a volumetric representation, by partitioning the space into equal-sized voxels.

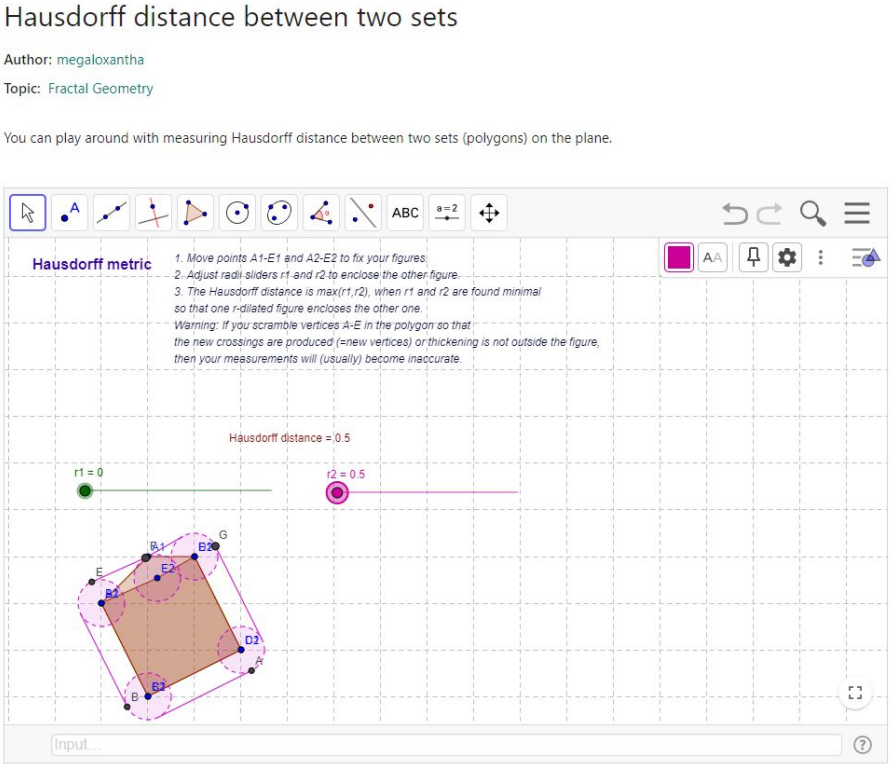
Suppose $f : \mathcal{X} \to \mathbb{R}$ is a continuous set function w.r.t Hausdorff distance $d_H(\cdot,\cdot)$. $\forall \epsilon > 0, \exists$ a continuous function h and a symmetric function $g(x_1, \ldots, x_n) = \gamma \circ MAX$, such that for any $S \in \mathcal{X}$,

$$\left| f(S) - \gamma \left(\underset{x_i \in S}{\mathrm{MAX}} \{h(x_i)\} \right) \right| < \epsilon$$

- MAX is a vector max operator that takes n vectors as input and returns a new vector of the element-wise maximum.
- $x_1, \ldots, x_n$ is the full list of elements in S ordered arbitrarily

Hausdorff distance between two sets

- Hausdorff Distance: The Hausdorff distance $d_H(S, S')$ between two sets S and $S\prime$ is a measure of the maximum distance of a set to the nearest point in the other set. It is defined as the greatest of all the distances from a point in one set to the closest point in the other set. The Hausdorff distance is commonly used to measure the dissimilarity between two sets and is particularly useful for comparing sets that are subsets of a metric space.



Proof

By the continuity of f, we take δ_ϵ so that $|\ f(S) - f\left(S'\right)\ | < \epsilon$ for any $S, S' \in \mathcal{X}$ if $d_H\left(S, S'\right) < \delta_\epsilon$.

Define $K = \lceil 1/\delta_\epsilon \rceil$, which split $[0,1]$ into K intervals evenly and define an auxiliary function that maps a point to the left end of the interval it lies in:

$$\sigma(x) = \frac{\lfloor Kx \rfloor}{K}$$

Let $\tilde{S} = \{\sigma(x) : x \in S\}$, then

$$|f(S) - f(\tilde{S})| < \epsilon$$

because $d_H(S, \tilde{S}) < 1/K \leq \delta_\epsilon$.

For $\delta_\epsilon = 0.25$ partition $[0,1]$ into $K = 4$ intervals evenly

- $[0, 0.25)$
- $[0.25, 0.5)$
- $[0.5, 0.75)$
- $[0.75, 1]$

Using $\sigma(x) = \frac{\lfloor Kx \rfloor}{K}$, we map each point in S to the left endpoint of its interval:

- $\sigma(0.1) = \frac{\lfloor 4 \times 0.1 \rfloor}{4} = \frac{0}{4} = 0$
- $\sigma(0.3) = \frac{\lfloor 4 \times 0.3 \rfloor}{4} = \frac{1}{4} = 0.25$
- $\sigma(0.7) = \frac{\lfloor 4 \times 0.7 \rfloor}{4} = \frac{2}{4} = 0.5$

So, $\tilde{S} = \{0, 0.25, 0.5\}$.

Proof

Let $h_k(x) = e^{-d\left(x, \left[\frac{k-1}{K}, \frac{k}{K}\right]\right)}$ be a soft indicator function where $d(x, I)$ is the point to set (interval) distance.

Let $\mathbf{h}(x) = [h_1(x); \ldots; h_K(x)]$, then $\mathbf{h} : \mathbb{R} \to \mathbb{R}^K$.

For each $x \in S$, let's say $\mathbf{h}(x)$ just indicates which interval x belongs to, with a 1 for occupancy and 0 otherwise (this is a simplification for illustrative purposes):

- $\mathbf{h}(0.1) = [1, 0, 0, 0]$
- $\mathbf{h}(0.3) = [0, 1, 0, 0]$
- $\mathbf{h}(0.7) = [0, 0, 1, 0]$

Let $v_j(x_1, \ldots, x_n) = \max\left\{\tilde{h}_j(x_1), \ldots, \tilde{h}_j(x_n)\right\}$, indicating the occupancy of the j-th interval by points in S. Let $\mathbf{v} = [v_1; \ldots; v_K]$, then $\mathbf{v} : \underbrace{\mathbb{R} \times \ldots \times \mathbb{R}}_{n} \to \{0, 1\}^K$ is a symmetric function, indicating the occupancy of each interval by points in S.

Given $\mathbf{h}(x)$ for each x, we find $\mathbf{v}$ by taking the max across these vectors, but since each x uniquely occupies its interval, $\mathbf{v}$ simply reflects their combined occupancy:

$$\mathbf{v} = [1, 1, 1, 0].$$

Proof

Define $\tau : \{0,1\}^K \to \mathcal{X}$ as $\tau(v) = \left\{\frac{k-1}{K} : v_k \geq 1\right\}$, which maps the occupancy vector to a set which contains the left end of each occupied interval. It is easy to show:

$$\tau\left(\mathbf{v}\left(x_1, \ldots, x_n\right)\right) \equiv \tilde{S}$$

where $x_1, \ldots, x_n$ are the elements of S extracted in certain order.

Applying τ to $\mathbf{v}$, we reconstruct a set that includes the left endpoints based on occupancy indicated by $\mathbf{v}$:

Since $\mathbf{v}$ indicates occupancy in the first three intervals,

$$\tau(\mathbf{v}) = \{0, 0.25, 0.5\},$$

$$\tilde{S} = \{0, 0.25, 0.5\}.$$

Let $\gamma : \mathbb{R}^K \to \mathbb{R}$ be a continuous function such that $\gamma(\mathbf{v}) = f(\tau(\mathbf{v}))$ for $v \in \{0,1\}^K$. Then,

$$\begin{aligned} &\left|\gamma\left(\mathbf{v}\left(x_1, \ldots, x_n\right)\right) - f(S)\right| \\ =&\left|f\left(\tau\left(\mathbf{v}\left(x_1, \ldots, x_n\right)\right)\right) - f(S)\right| < \epsilon \end{aligned}$$

Note that $\gamma\left(\mathbf{v}\left(x_1, \ldots, x_n\right)\right)$ can be rewritten as follows:

$$\begin{aligned} \gamma\left(\mathbf{v}\left(x_1, \ldots, x_n\right)\right) &= \gamma\left(\mathrm{MAX}\left(\mathbf{h}\left(x_1\right), \ldots, \mathbf{h}\left(x_n\right)\right)\right) \\ &= \left(\gamma \circ \mathrm{MAX}\right)\left(\mathbf{h}\left(x_1\right), \ldots, \mathbf{h}\left(x_n\right)\right) \end{aligned}$$

Obviously $\gamma \circ MAX$ is a symmetric function.

Theoretical Analysis : Bottleneck dimension and stability

- Intuitively, our network learns to summarize a shape by a sparse set of key points

Suppose $\mathbf{u} : \mathcal{X} \to \mathbb{R}^K$ such that $\mathbf{u} = \underset{x_i \in S}{\mathrm{MAX}} \{h(x_i)\}$ and $f = \gamma \circ \mathbf{u}$.

- $\mathbf{u} = \underset{x_i \in S}{\mathrm{MAX}} \{h(x_i)\}$ is the sub-network of f which maps a point set in $[0,1]^m$ to a K-dimensional vector.

Then:

(a) $\forall S, \exists \mathcal{C}_S, \mathcal{N}_S \subseteq \mathcal{X}, f(T) = f(S)$ if $\mathcal{C}_S \subseteq T \subseteq \mathcal{N}_S$;

- $f(S)$ is unchanged up to the input corruption if all points in $\mathcal{C}_S$ are preserved; it is also unchanged with extra noise points up to $\mathcal{N}_S$.

(b) $|\mathcal{C}_S| \leq K$

- $\mathcal{C}_S$ only contains a bounded number of points, determined by K
- $f(S)$ is in fact totally determined by a finite subset $\mathcal{C}_S \subseteq S$ of less or equal to K elements. We therefore call $\mathcal{C}_S$ the critical point set of S and K the bottleneck dimension of f

Applications : 3D Object Classification

ModelNet40 - 12,311 CAD models from 40 man-made object categories

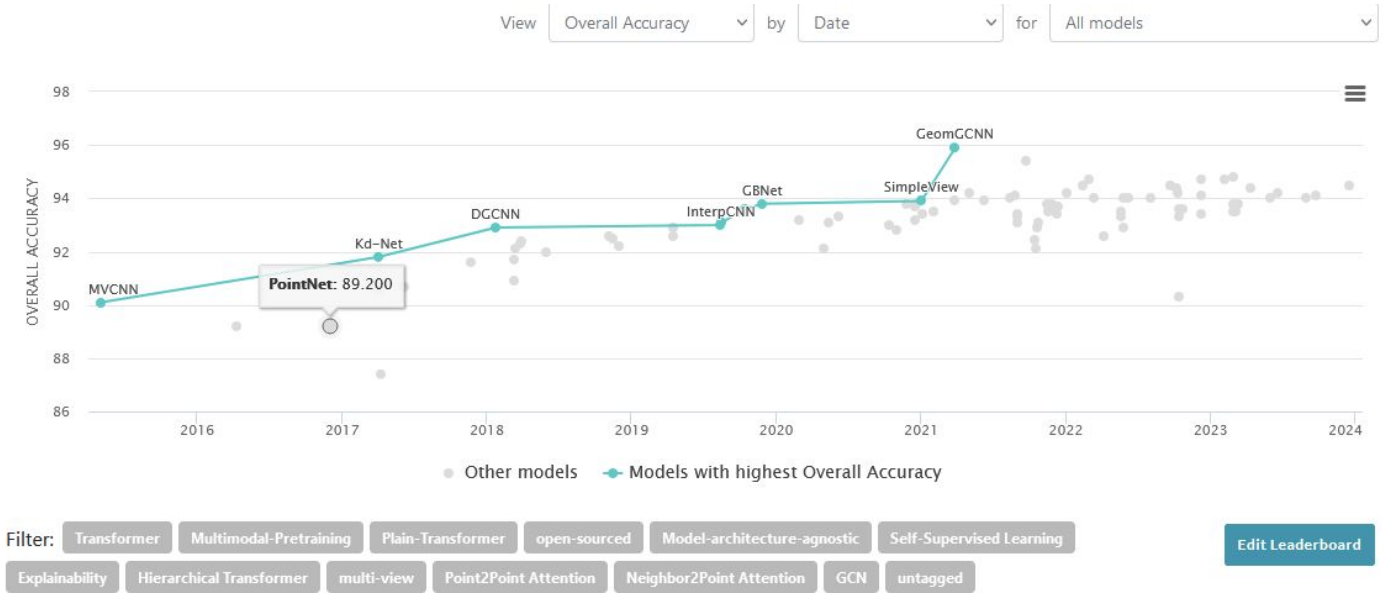
- 9,843 training
- 2,468 testing

	input	#views	accuracy avg. class	accuracy overall
SPH [11]	mesh	-	68.2	-
3DShapeNets [28]	volume	1	77.3	84.7
VoxNet [17]	volume	12	83.0	85.9
Subvolume [18]	volume	20	86.0	**89.2**
LFD [28]	image	10	75.5	-
MVCNN [23]	image	80	**90.1**	-
Ours baseline	point	-	72.6	77.4
Ours PointNet	point	1	86.2	**89.2**

Table 1. **Classification results on ModelNet40.** Our net achieves state-of-the-art among deep nets on 3D input.

Applications : 3D Object Classification

[ModelNet40]



Applications : 3D Object Part Segmentation

ShapeNet - 16,881 shapes from 16 categories, annotated with 50 parts in total. Most object categories are labeled with two to five parts.

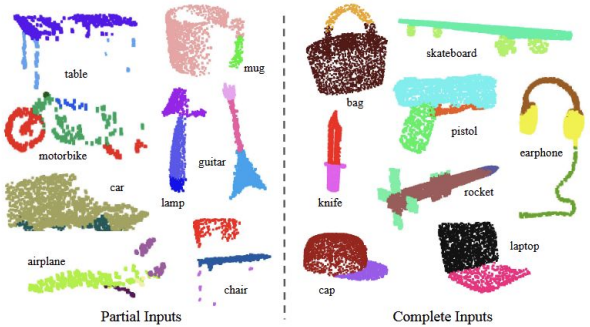
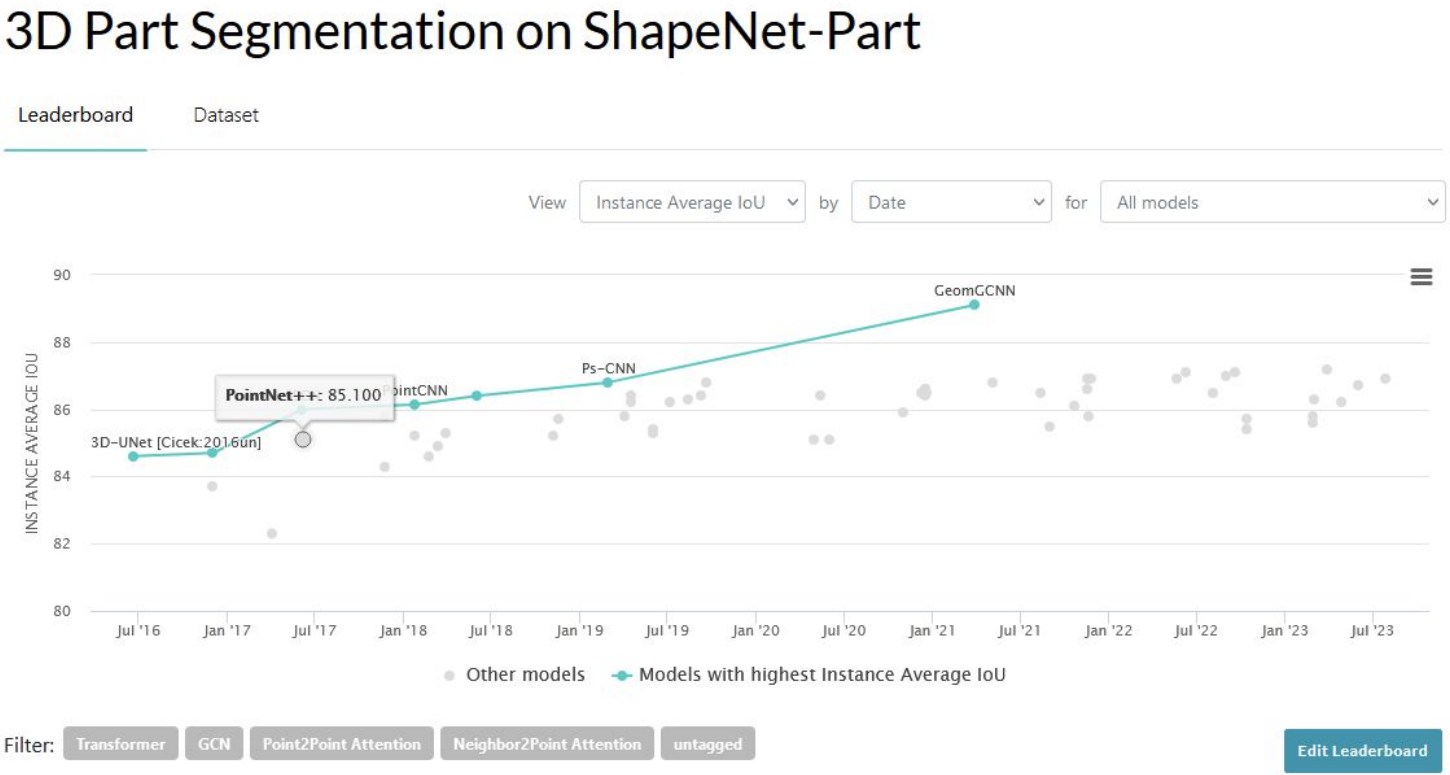


Figure 3. **Qualitative results for part segmentation.** We visualize the CAD part segmentation results across all 16 object categories. We show both results for partial simulated Kinect scans (left block) and complete ShapeNet CAD models (right block).

	mean	aero	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	skate board	table
# shapes		2690	76	55	898	3758	69	787	392	1547	451	202	184	283	66	152	5271
Wu [27]	-	63.2	-	-	-	73.5	-	-	-	74.4	-	-	-	-	-	-	74.8
Yi [29]	81.4	81.0	78.4	77.7	**75.7**	87.6	61.9	**92.0**	85.4	**82.5**	**95.7**	**70.6**	91.9	**85.9**	53.1	69.8	75.3
3DCNN	79.4	75.1	72.8	73.3	70.0	87.2	63.5	88.4	79.6	74.4	93.9	58.7	91.8	76.4	51.2	65.3	77.1
Ours	**83.7**	**83.4**	**78.7**	**82.5**	74.9	**89.6**	**73.0**	91.5	**85.9**	80.8	95.3	65.2	**93.0**	81.2	**57.9**	**72.8**	**80.6**

Table 2. **Segmentation results on ShapeNet part dataset.** Metric is mIoU(%) on points. We compare with two traditional methods [27] and [29] and a 3D fully convolutional network baseline proposed by us. Our PointNet method achieved the state-of-the-art in mIoU.

Applications : 3D Object Part Segmentation



Applications : Semantic Segmentation in Scenes

Stanford 3D semantic parsing dataset - 3D scans from Matterport scanners in 6 areas including 271 rooms. Each point in the scan is annotated with one of the semantic labels from 13 categories

	mean IoU	overall accuracy
Ours baseline	20.12	53.19
Ours PointNet	**47.71**	**78.62**

Table 3. **Results on semantic segmentation in scenes.** Metric is average IoU over 13 classes (structural and furniture elements plus clutter) and classification accuracy calculated on points.

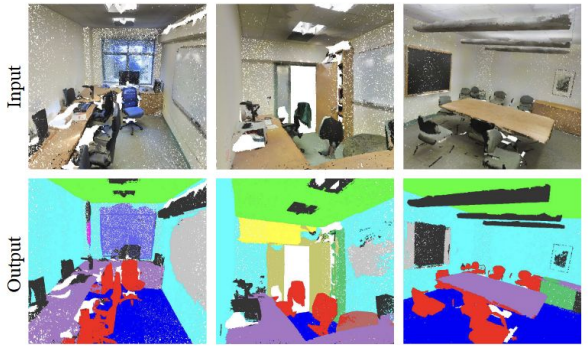
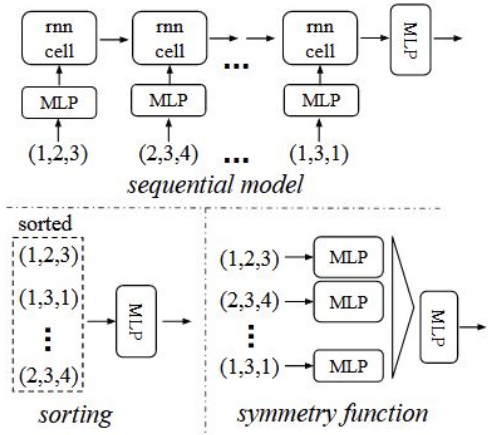


Figure 4. **Qualitative results for semantic segmentation.** Top row is input point cloud with color. Bottom row is output semantic segmentation result (on points) displayed in the same camera viewpoint as input.

Experiments : Comparison with Alternative Order-invariant Methods



	accuracy
MLP (unsorted input)	24.2
MLP (sorted input)	45.0
LSTM	78.5
Attention sum	83.0
Average pooling	83.8
Max pooling	**87.1**

Figure 5. **Three approaches to achieve order invariance.** Multi-layer perceptron (MLP) applied on points consists of 5 hidden layers with neuron sizes 64,64,64,128,1024, all points share a single copy of MLP. The MLP close to the output consists of two layers with sizes 512,256.

Experiments : Effectiveness of Input and Feature Transformations

Transform	accuracy
none	87.1
input (3x3)	87.9
feature (64x64)	86.9
feature (64x64) + reg.	87.4
both	**89.2**

Table 5. **Effects of input feature transforms.** Metric is overall classification accuracy on ModelNet40 test set.

Experiments : Robustness Test

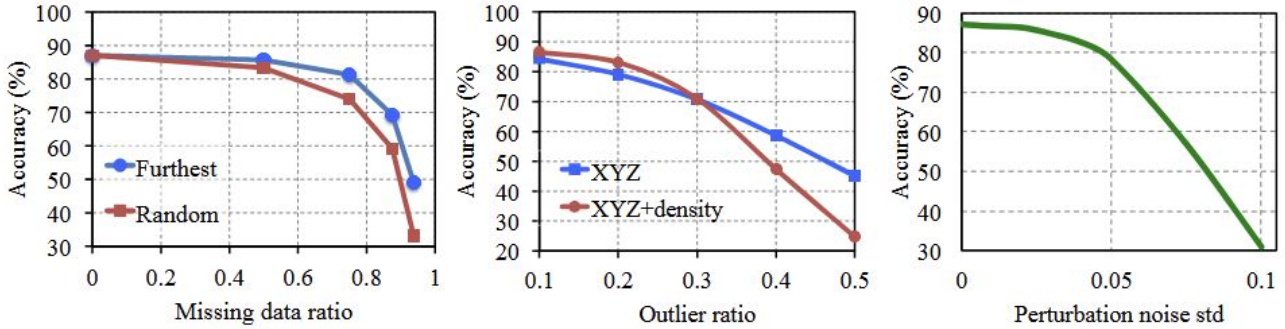


Figure 6. **PointNet robustness test.** The metric is overall classification accuracy on ModelNet40 test set. Left: Delete points. Furthest means the original 1024 points are sampled with furthest sampling. Middle: Insertion. Outliers uniformly scattered in the unit sphere. Right: Perturbation. Add Gaussian noise to each point independently.

Experiments : Visualizing PointNet

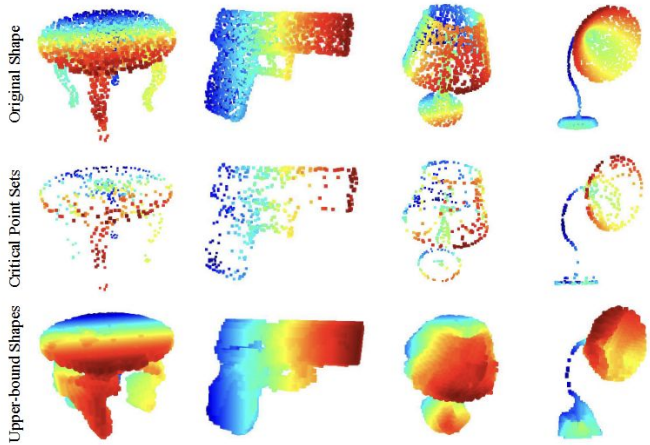


Figure 7. **Critical points and upper bound shape.** While critical points jointly determine the global shape feature for a given shape, any point cloud that falls between the critical points set and the upper bound shape gives exactly the same feature. We color-code all figures to show the depth information.

- visualization of $\mathcal{C}_S$ critical point set and $\mathcal{N}_S$ upper-bound set. Both reflect the robustness of PointNet
  - any set of points T you could form by adding points to $\mathcal{C}_S$ or removing points from $\mathcal{N}_S$ will give exactly the same global shape feature $f(S)$.
- $\mathcal{C}_S$ summarizes the skeleton of the shape.
- $\mathcal{N}_S$ illustrates the largest possible point cloud that give the same global shape feature $f(S)$ as the input point cloud S

PointNet++

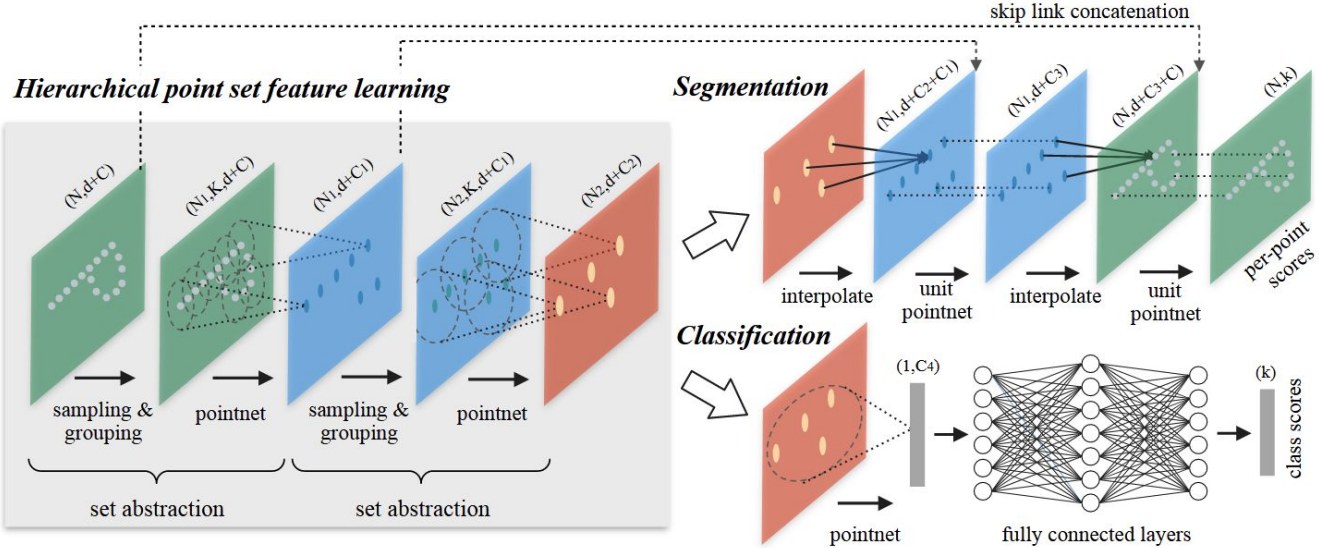
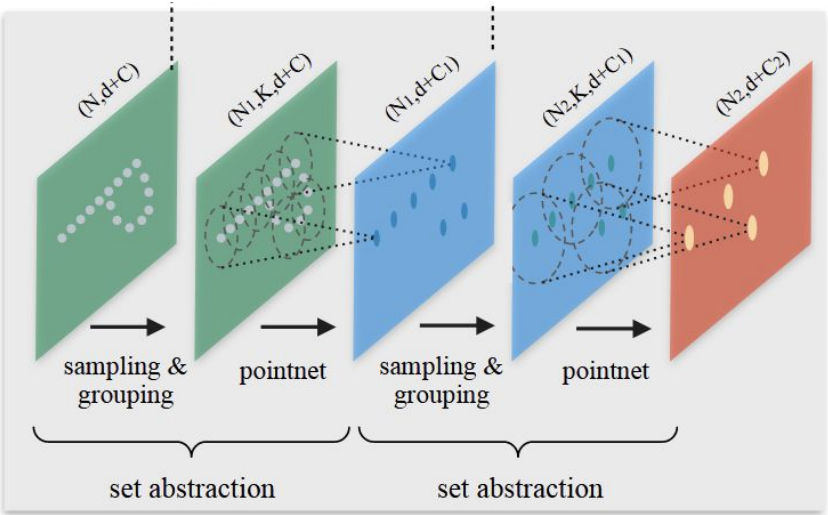


Figure 2: Illustration of our hierarchical feature learning architecture and its application for set segmentation and classification using points in 2D Euclidean space as an example. Single scale point grouping is visualized here. For details on density adaptive grouping, see Fig. 3

Hierarchical Point Set Feature Learning

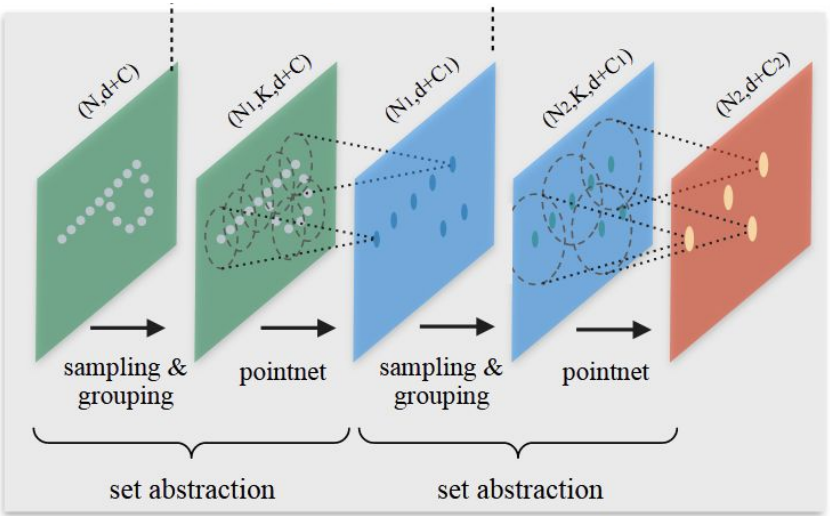
While PointNet uses a single max pooling operation to aggregate the whole point set, PointNet++ builds a hierarchical grouping of points and progressively abstract larger and larger local regions along the hierarchy.



set abstraction levels

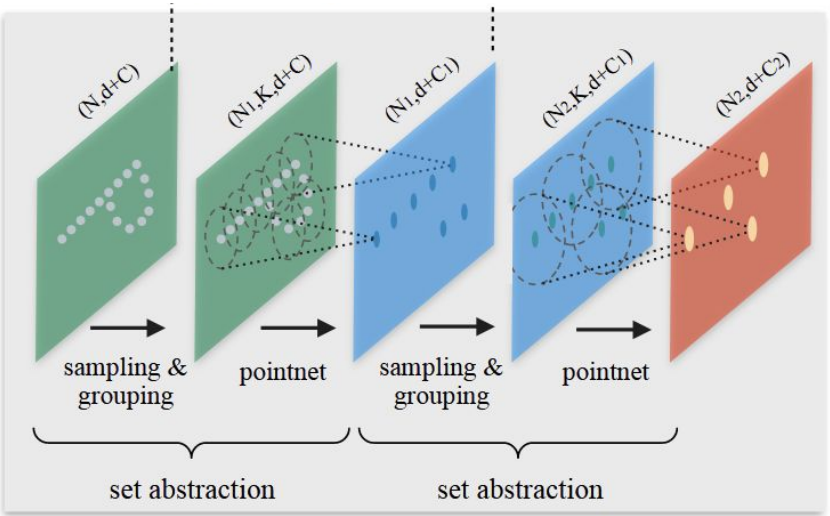
- input: $N \times (d + C)$ matrix
  - N points with
  - $d - dim$ coordinates
  - $C - dim$ point feature.
- outputs: $N' \times (d + C')$ matrix
  - N' subsampled points
  - $d - dim$ coordinates
  - $C\prime$-dim feature vectors summarizing local context

Hierarchical Point Set Feature Learning



- Sampling layer selects points as centroids
  - iterative farthest point sampling (FPS) to choose a subset of points
- Grouping layer finds neighbor points to the centroid
  - input: pointset $N \times (d + C)$ and centroids $N' \times d$
  - output: groups of point sets of size $N' \times K \times (d + C)$,
    - Ball query finds all points that are within a radius to the query point with an upper limit of K
- PointNet layer encodes the points into features
  - input: local regions of points with data size $N' \times (d + C)$
  - output: $N' \times (d + C')$
  - The coordinates of points in a local region are firstly translated into a local frame relative to the centroid point

Hierarchical Point Set Feature Learning



- Sampling layer selects points as centroids
  - iterative farthest point sampling (FPS) to choose a subset of points
- Grouping layer finds neighbor points to the centroid
  - input: pointset $N \times (d + C)$ and centroids $N' \times d$
  - output: groups of point sets of size $N' \times K \times (d + C)$,
    - Ball query finds all points that are within a radius to the query point with an upper limit of K
- PointNet layer encodes the points into features
  - input: local regions of points with data size $N' \times (d + C)$
  - output: $N' \times (d + C')$
  - The coordinates of points in a local region are firstly translated into a local frame relative to the centroid point

Non-Uniform Sampling Density

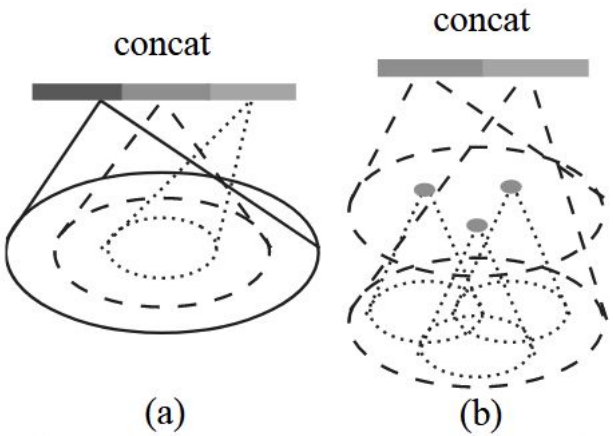


Figure 3: (a) Multi-scale grouping (MSG); (b) Multi-resolution grouping (MRG).

Point sets come with nonuniform density in different areas. This requires inspecting closely in dense regions and in large scale and low density areas.

PointNet++ layers are PointNet layers that learn to combine features from regions of different scales when the input sampling density changes.

Multi-scale grouping

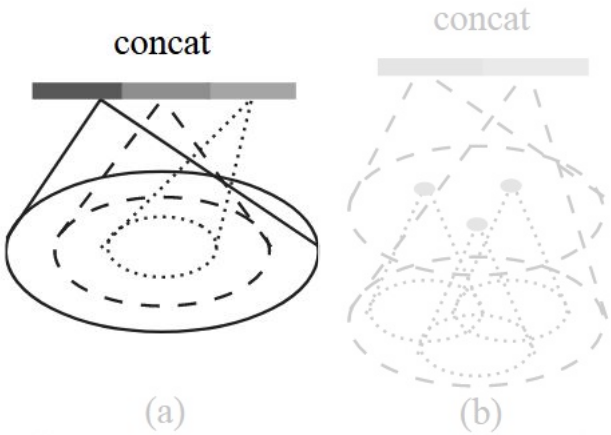


Figure 3: (a) Multi-scale grouping (MSG); (b) Multi-resolution grouping (MRG).

Apply grouping layers with different scales followed by according PointNets to extract features of each scale. Features at different scales are concatenated to form a multi-scale feature.

Optimize by randomly dropping out input points with a randomized probability for each instance,

Multi-resolution grouping

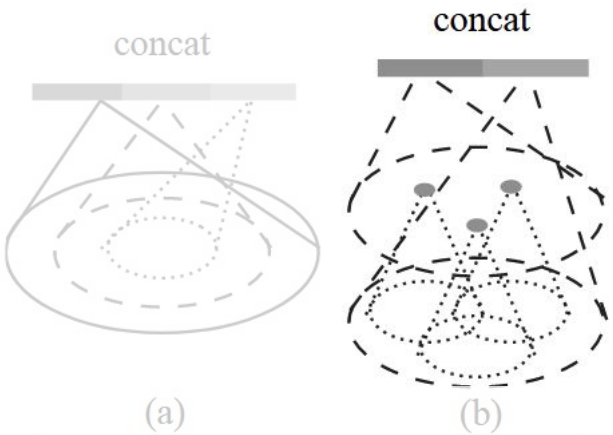


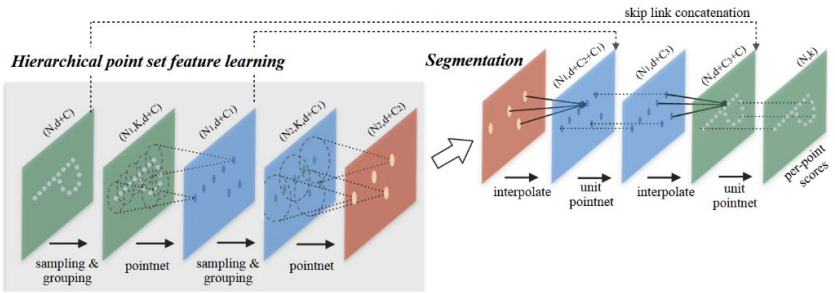
Figure 3: (a) Multi-scale grouping (MSG); (b) Multi-resolution grouping (MRG).

Features of a region at some level is a concatenation of two vectors:

- One vector (left in figure) is obtained by summarizing the features at each subregion from the lower level using the set abstraction level.
- The other vector (right) is the feature that is obtained by directly processing all raw points in the local region using a single PointNet.

Point Feature Propagation for Set Segmentation

To obtain point features for all the original points Pointnet++ uses a hierarchical propagation strategy with distance based interpolation and across level skip links.



- In a feature propagation level, we propagate point features from $N_l \times (d + C)$ points to N_{l-1} points where N_{l-1} and N_l (with $N_l \leq N_{l-1}$) are point set sizes of input and output of set abstraction level l.
- The interpolated features on N_{l-1} points are then concatenated with skip-linked point features from the set abstraction level.
- The concatenated features are passed through a "unit pointnet", which is similar to one-by-one convolution in CNNs. A few shared fully connected and ReLU layers are applied to update each point's feature vector.

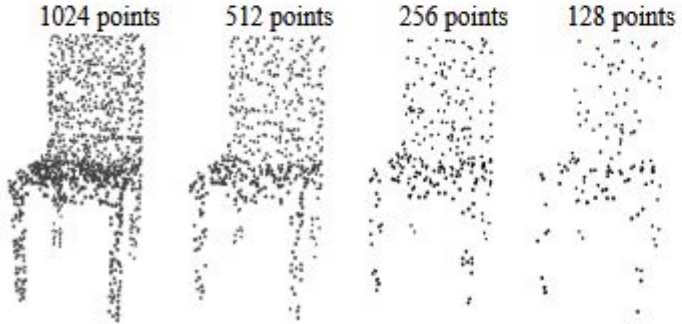
Point Set Classification in Euclidean Metric Space

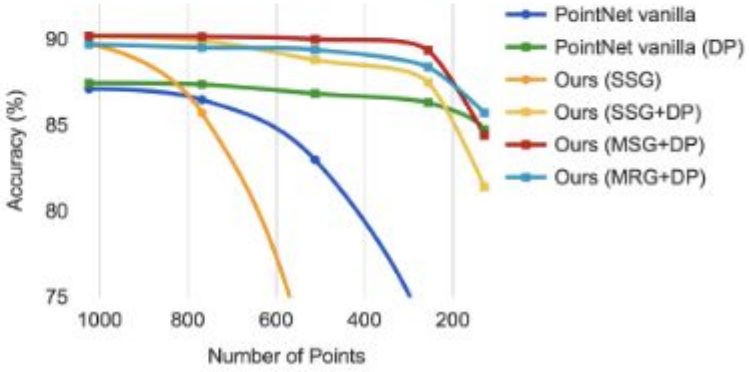
Method	Error rate (%)
Multi-layer perceptron [24]	1.60
LeNet5 [11]	0.80
Network in Network [13]	**0.47**
PointNet (vanilla) [20]	1.30
PointNet [20]	0.78
Ours	0.51

Table 1: MNIST digit classification.

Method	Input	Accuracy (%)
Subvolume [21]	vox	89.2
MVCNN [26]	img	90.1
PointNet (vanilla) [20]	pc	87.2
PointNet [20]	pc	89.2
Ours	pc	90.7
Ours (with normal)	pc	**91.9**

Table 2: ModelNet40 shape classification.





Point Set Segmentation for Semantic Scene Labeling

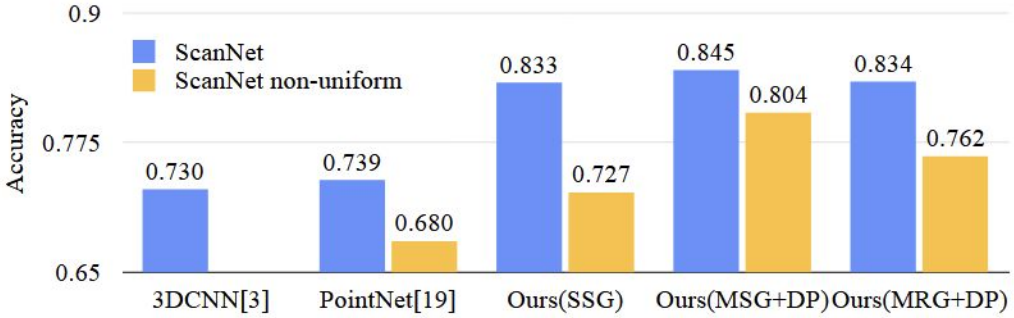


Figure 5: Scannet labeling accuracy.

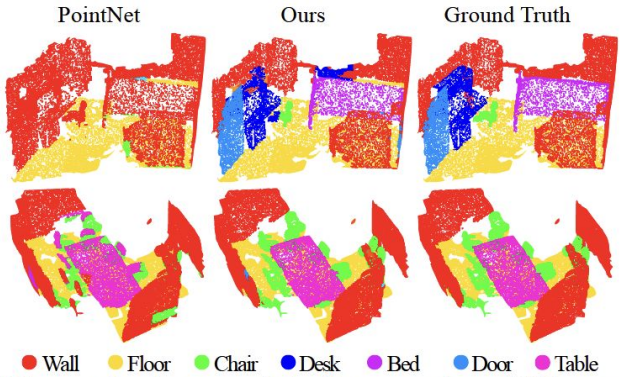


Figure 6: Scannet labeling results. [20] captures the overall layout of the room correctly but fails to discover the furniture. Our approach, in contrast, is much better at segmenting objects besides the room layout.