Point set-based modalities. Invariance and equivariance in learning

Geometric Computer Vision

GCV v2021.1, Module 4

Alexey Artemov, Spring 2021

Lecture Outline

Lecture Outline

§1. 3D point clouds and their properties

Depth images and point clouds

1.1. Depth images and point clouds

§1. 3D point clouds and their properties

- **Depth images:** 2D images, where each pixel stores depth value
  - $I : \Omega \to \mathbb{R}_+, \quad \Omega \subset \mathbb{R}^2$
  - Need multiple depth images to capture the full scene or object
- **Point clouds:** a set of points in 3D space
  - $P = \{\mathbf{p}_i\}_{i=1}^n \subset \mathbb{R}^3$
  - Obtained **directly** (e.g., laser scanning) or by **combining depth images** (e.g. registration)

1.1. Depth images and point clouds

§1. 3D point clouds and their properties

- Depth image ↔ point cloud: re-projection 2D ↔ 3D (need to know camera intrinsics!)

```
Point cloud =
  unpixelize(
    unproject(
      depth image))
```

- Perspective projection K_f, image to pixel transform K_s: $P = K^{-1}I = K_f^{-1} \underbrace{K_s^{-1} \circ I}_{(w,h)\to(3,n)}$

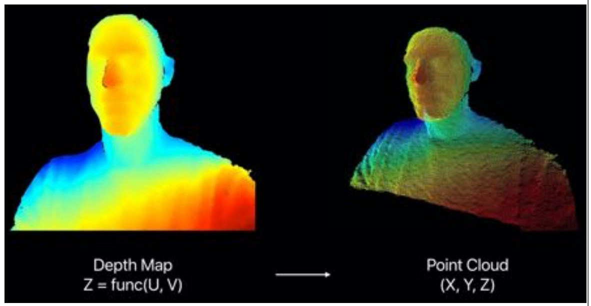


Figure credit: idownloadblog.com

1.1. Point clouds: geometric properties

§1. 3D point clouds and their properties

- **Sampling density:** distribution of points on the surface
- **Noise:** distribution of points **near** the surface
- **Outliers:** points far from the true surface
- **Misalignment:** poor registration of range-images
- **Missing data:** portions of the shape not being sampled

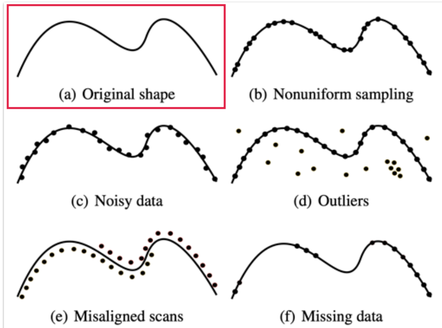
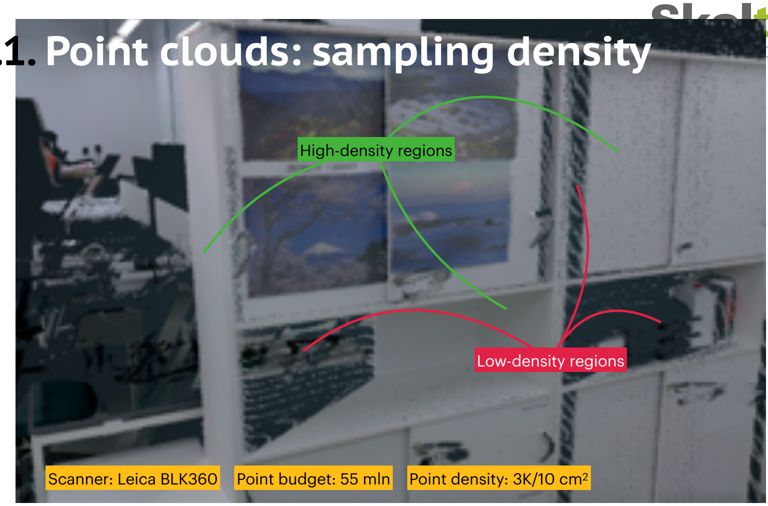


Figure 2: *Different forms of point cloud artifacts, shown here in the case of a curve in 2D.*

Figure credit: Berger, M., Tagliasacchi, A., Seversky, L.M., Alliez, P., Guennebaud, G., Levine, J.A., Sharf, A. and Silva, C.T., 2017, January. A survey of surface reconstruction from point clouds. In *Computer Graphics Forum* (Vol. 36, No. 1, pp. 301-329).

1.1. Point clouds: sampling density



1.1. Point clouds: noise

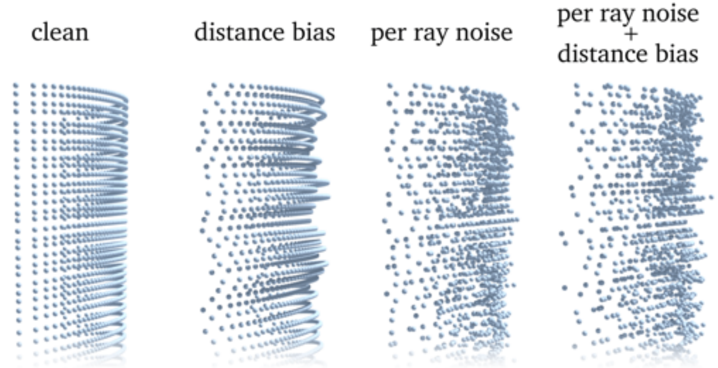


Figure 13: *Simulated noise of a Velodyne HDL-64E scanner. Here*

Figure credit: PointCleanNet

1.1. Point clouds: outliers

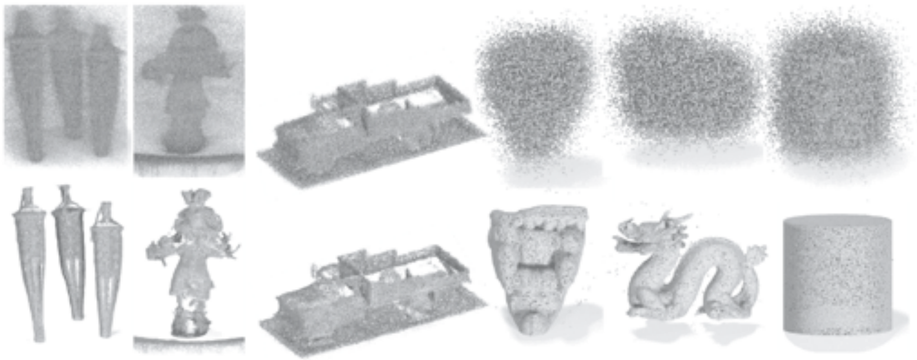


Figure credit: PointCleanNet

1.1. Point clouds: misalignment

Figure credit: peel-3d

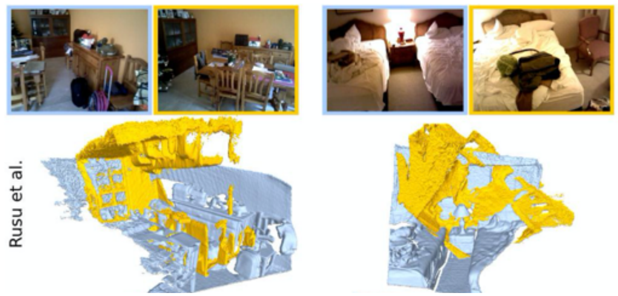


Figure credit: 3DMatch

1.1. Point clouds: missing data

Figure credit: PCN: Point Completion Network

Recap: invariance and equivariance

1.2. Properties of real-world problems

§1. 3D point clouds and their properties

- Detecting cars in aerial images
- Orientation does not matter (?)
- Classification: is the car present in the image? → invariance
- Segmentation: produce car masks → equivariance
- Angle prediction: output depends on the input in a predefined way → covariance

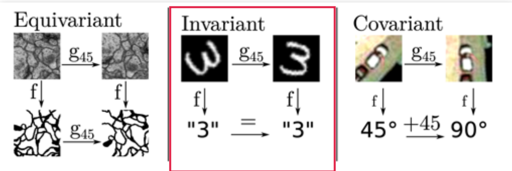


Figure 1. Desirable behaviors with respect to rotation of the inputs: (left) equivariance in segmentation; (center) invariance in classification; (right) covariance in absolute orientation estimation. g_{45} is an operator that rotates the input image by 45°.

Figure credit: Rotation equivariant vector field networks

1.2. Invariance

§1. 3D point clouds and their properties

- $f(\cdot) : X \to Y$: a function with domain X and codomain Y
- $T(\cdot) : X \to X$: a transformation of the input space (e.g., translation, rotation…)
- f is invariant w.r.t. the transformation T if: $f(T(x)) = f(x), \quad \forall x \in X$

- Commonly wanted: invariance to rotations when classifying images (e.g. aerial, biological, …)

1.2. Equivariance

§1. 3D point clouds and their properties

- $f(\cdot) : X \to Y$: a function with domain X and codomain Y
- $T^X(\cdot) : X \to X, T^Y(\cdot) : Y \to Y$: transformations of the domain and codomain
- f is equivariant w.r.t. the transformation T if: $f(T^X(x)) = T^Y(f(x)), \quad \forall x \in X$
- Equivariance is a generalization of invariance: if $T^Y(\cdot) = \mathrm{id}$, then equivariance expresses invariance
- Many tasks in vision: segmentation, edge/contour prediction

Properties of Point Sets in $\mathbb{R}^3$

1.2. Point Sets in $\mathbb{R}^3$

§1. 3D point clouds and their properties

- **Unordered (set structure):** 3D point clouds cannot be canonically ordered
  - Different from (range-)images, voxels…
- **Interactions between points:** local (surface) geometry of 3D point clouds
  - Need to capture local structure of point sets
- **Invariance under transformations:** rotations and translations in 3D do not change global object category

Set structure: permutations

1.3. Point Sets in $\mathbb{R}^3$: permutations

Skoltech
Skolkovo Institute of Science and Technology

§1. 3D point clouds and their properties

- $f(\cdot) : X \to Y$: a function with domain X and codomain Y
- Input is a set X → the input domain is a power set $X \to 2^X$
- Define a **permutation:** a bijective function acting on sets of indexes $\pi(\cdot) : \{1, \dots, M\} \to \{1, \dots, M\}$
- **Permutation invariance:** the function $f(\cdot) : 2^X \to Y$ is permutation invariant if for any permutation π:

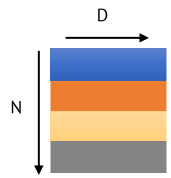
$$f(\{x_1, \dots, x_M\}) = f(\{x_{\pi(1)}, \dots, x_{\pi(M)}\})$$

1.3. Point Sets in $\mathbb{R}^3$: unordered input

Skoltech
Skolkovo Institute of Science and Technology

§1. 3D point clouds and their properties

Point cloud: N **orderless** points, each represented by a D dim vector

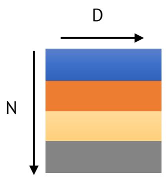


1.3. Point Sets in $\mathbb{R}^3$: unordered input

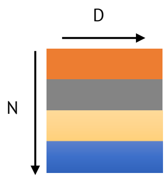
Skoltech
Skolkovo Institute of Science and Technology

§1. 3D point clouds and their properties

Point cloud: N **orderless** points, each represented by a D dim vector



represents the same **set** as

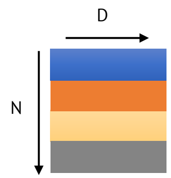


1.3. Point Sets in $\mathbb{R}^3$: unordered input

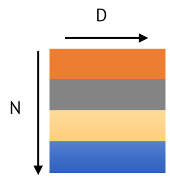
Skoltech
Skolkovo Institute of Science and Technology

§1. 3D point clouds and their properties

Point cloud: N **orderless** points, each represented by a D dim vector



represents the same **set** as



Model needs to be invariant to N! permutations

1.3. Point Sets in $\mathbb{R}^3$: permutations

Skoltech
Skolkovo Institute of Science and Technology

§1. 3D point clouds and their properties

- **Theorem (DeepSets):**
  "A function $f(\cdot)$ operating on a set X having elements from a countable universe, is a valid set function, i.e., **invariant to the permutation** of instances in X, iff it can be decomposed in the form

$$f(x) = \rho\Big(\sum_{x \in X} \phi(x)\Big)$$

  for suitable transformations ϕ and ρ."
- **Interpretation:** element-wise transformation (ϕ) following by aggregation (ρ).

Zaheer, M., Kottur, S., Ravanbhakhsh, S., Póczos, B., Salakhutdinov, R., & Smola, A. J. (2017, December). Deep Sets. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (pp. 3394-3404).

Skoltech
Skolkovo Institute of Science and Technology

Geometric structure: rotation, translation

1.4. Point Sets in $\mathbb{R}^3$: rotation/translation

Skoltech
Skolkovo Institute of Science and Technology

§1. 3D point clouds and their properties

- Recap from the last lecture: common transformations experiences by objects in $\mathbb{R}^3$ are **3D rotations and translations**
- 3D rotation: defined by 3×3 orthogonal matrix $R \in \mathrm{SO}(3)$ where $\mathrm{SO}(3)$ is the special orthogonal group: $\mathrm{SO}(3) = \{R \in \mathbb{R}^{3\times 3}, R^T R = 1, \det R = +1\}$
- 3D translation: defined by 3×1 column-vector T
- Rigid-body motion (homogeneous coordinates): $X = RX_0 + T = g(X_0)$ where g is defined by $g = \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix}$
- Commonly wanted: invariance to rotations R, or roto-translations g

1.2. Point Sets in $\mathbb{R}^3$: summary

Skoltech
Skolkovo Institute of Science and Technology

§1. 3D point clouds and their properties

- Unordered (invariant under permutations)
- Invariance under roto-translation transformations
- Local (surface) geometry of 3D point clouds
- Deep learning models need to capture these properties!

Skoltech
Skolkovo Institute of Science and Technology

§2. Constructing neural networks on point clouds

Skoltech
Skolkovo Institute of Science and Technology

Building invariances into point-based neural networks

2.1. PointNets: challenges

Skoltech
Skolkovo Institute of Science and Technology

§2. Constructing neural networks on point clouds

- Can we achieve effective **feature learning directly** on point clouds?

- **Unordered point set as input**
  - Model needs to be invariant to N! permutations.

- Invariance under geometric transformations
  - Point cloud rotations should not alter classification results.

2.1. Permutation invariance: symmetric functions

Skoltech
Skolkovo Institute of Science and Technology

§2. Constructing neural networks on point clouds

$$f(\{x_1, \dots, x_M\}) = f(\{x_{\pi(1)}, \dots, x_{\pi(M)}\}), \quad x_i \in \mathbb{R}^D$$

2.1. Permutation invariance: symmetric functions

§2. Constructing neural networks on point clouds

$$f(\{x_1, \ldots, x_M\}) = f(\{x_{\pi(1)}, \ldots, x_{\pi(M)}\}), \quad x_i \in \mathbb{R}^D$$

Examples:

$$f(x_1, x_2, \ldots, x_M) = \max\{x_1, x_2, \ldots, x_M\}$$

$$f(x_1, x_2, \ldots, x_M) = x_1 + x_2 + \ldots + x_M$$

…

2.1. Permutation invariance: symmetric functions

§2. Constructing neural networks on point clouds

$$f(\{x_1, \ldots, x_M\}) = f(\{x_{\pi(1)}, \ldots, x_{\pi(M)}\}), \quad x_i \in \mathbb{R}^D$$

Examples:

$$f(x_1, x_2, \ldots, x_M) = \max\{x_1, x_2, \ldots, x_M\}$$

$$f(x_1, x_2, \ldots, x_M) = x_1 + x_2 + \ldots + x_M$$

…

How can we construct a family of symmetric functions by neural networks?

2.1. Permutation invariance: symmetric functions

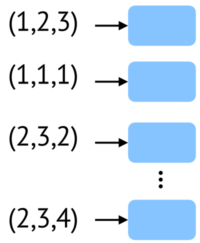
§2. Constructing neural networks on point clouds

$f(x_1, x_2, \ldots, x_M) = \gamma \circ g(h(x_1), \ldots, h(x_M))$ is symmetric if g is symmetric

2.1. Permutation invariance: symmetric functions

§2. Constructing neural networks on point clouds

$f(x_1, x_2, \ldots, x_M) = \gamma \circ g(h(x_1), \ldots, h(x_M))$ is symmetric if g is symmetric

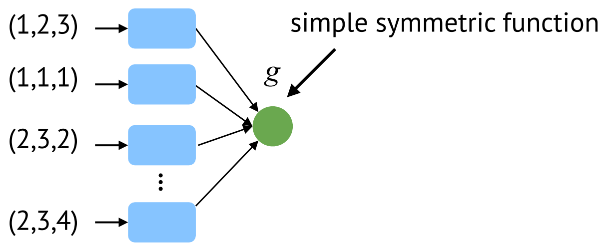


2.1. Permutation invariance: symmetric functions

Skoltech

§2. Constructing neural networks on point clouds

$f(x_1, x_2, \ldots, x_M) = \gamma \circ g(h(x_1), \ldots, h(x_M))$ is symmetric if g is symmetric

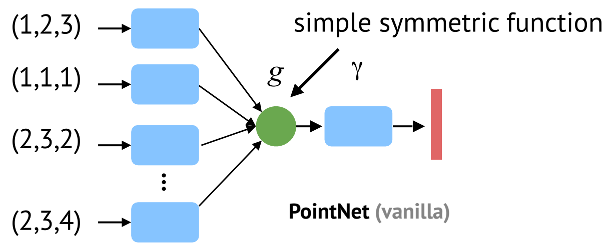


2.1. Permutation invariance: symmetric functions

Skoltech

§2. Constructing neural networks on point clouds

$f(x_1, x_2, \ldots, x_M) = \gamma \circ g(h(x_1), \ldots, h(x_M))$ is symmetric if g is symmetric

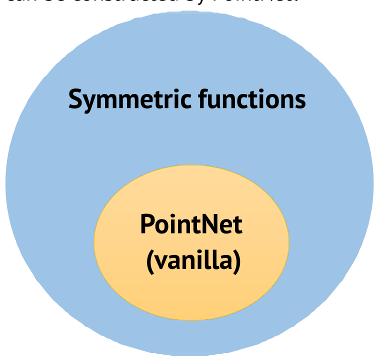


2.1. Permutation invariance: symmetric functions

Skoltech

§2. Constructing neural networks on point clouds

What symmetric functions can be constructed by PointNet?



2.1. Universal set function approximator

Skoltech

§2. Constructing neural networks on point clouds

- **Theorem (PointNet):**

Let $f(\cdot): 2^X \to \mathbb{R}$ be a continuous set function w.r.t Hausdorff distance $d_H(\cdot,\cdot)$. Then, it can be arbitrarily approximated by PointNet, meaning that $\forall S \in 2^X, S = (x_1, \ldots, x_M)$, and $\forall \epsilon > 0, \quad \exists h(\cdot), g(S) = \gamma \circ MAX$

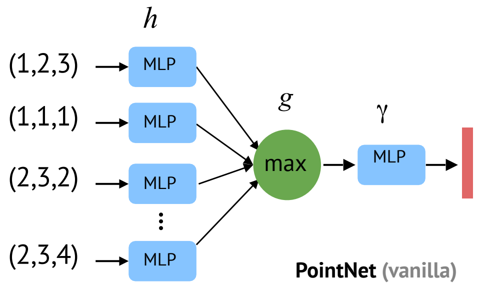
$$\left| f(S) - \boxed{\gamma \left(\underset{x_i \in S}{\mathrm{MAX}} \{h(x_i)\} \right)} \right| < \epsilon$$

PointNet (vanilla)

2.1. Basic PointNet Architecture

§2. Constructing neural networks on point clouds

- Empirically, use **multi-layer perceptron (MLP)** and **max pooling**:



PointNet (vanilla)

2.1. More permutation-invariant NNs

§2. Constructing neural networks on point clouds

- PointNet family or architectures (right):
  - [PointNet++](), [DGCNN](), [PointWeb](), …
- Geometric to regular representation:
  - [KPConv](), [PointwiseCNN]()
- Many implementations available in PyTorch-Geometric library – an extention to PyTorch

PyTorch geometric

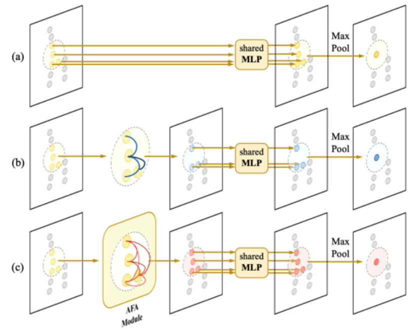


Figure Credit: PointWeb: Enhancing Local Neighborhood Features for Point Cloud Processing

2.1. PointNets: challenges

§2. Constructing neural networks on point clouds

- Can we achieve effective **feature learning directly** on point clouds?

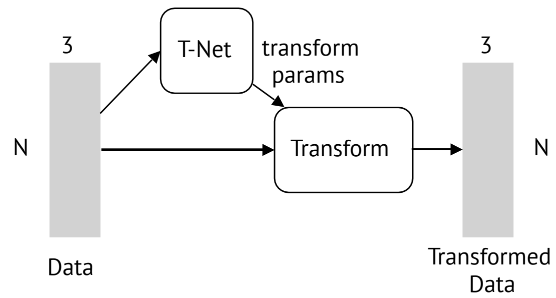
- Unordered point set as input
  - Model needs to be invariant to N! permutations.

- **Invariance under geometric transformations**
  - Point cloud rotations should not alter classification results.

2.1. Input alignment by transformer network

§2. Constructing neural networks on point clouds

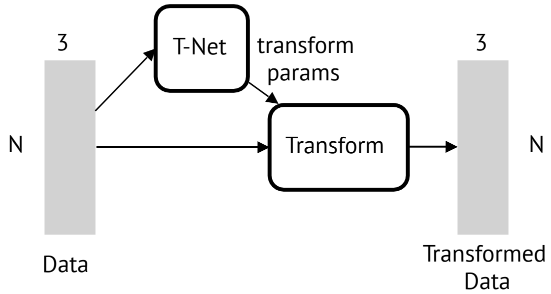
- Idea: Data dependent transformation for automatic alignment



2.1. Input alignment by transformer network

§2. Constructing neural networks on point clouds

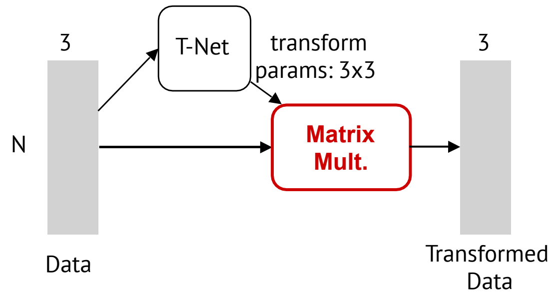
- Idea: Data dependent transformation for automatic alignment



2.1. Input alignment by transformer network

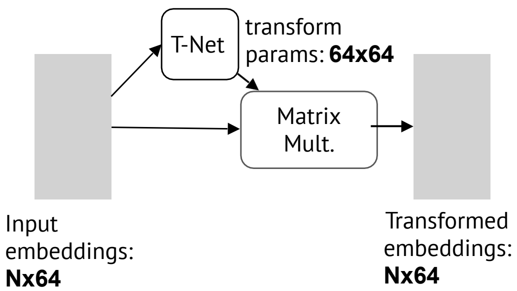
§2. Constructing neural networks on point clouds

- The transformation is just matrix multiplication!



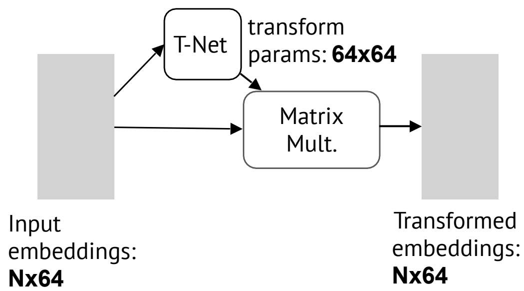
2.1. Embedding space alignment

§2. Constructing neural networks on point clouds



2.1. Embedding space alignment

§2. Constructing neural networks on point clouds



Regularization:

Transform matrix A 64x64 close to orthogonal:

$$L_{reg} = \|I - AA^T\|_F^2$$

2.1. Invariant point-based NNs: recap

§2. Constructing neural networks on point clouds

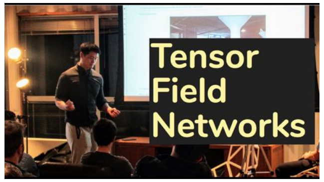
- **Permutation invariance:** use mappings of the form $f(x_1, \ldots, x_M) = \gamma \circ MAX_i(\{h(x_i)\})$, approximate γ and h using MLPs
- **Rotation invariance:** align to a local canonical orientation using a learned transformation

2.1. More equivariant and invariant NNs

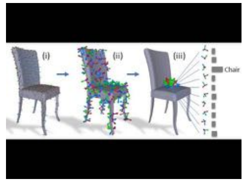
§2. Constructing neural networks on point clouds

- **In search of equivariant architectures:** much attention (and progress!) in recent years
- Check GitHub links and clickable videos for yourself to get familiar

SE(3)-Transformers: 3D Roto-Translation Equivariant Attention Networks



Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds.



Quaternion Equivariant Capsule Networks for 3D Point Clouds

Neural architecture: PointNet

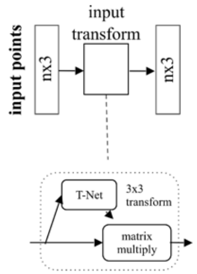
2.2. PointNet classification network

§2. Constructing neural networks on point clouds

Slide credit: Charles R. Qi (Stanford)

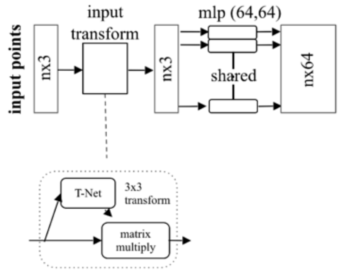
2.2. PointNet classification network

§2. Constructing neural networks on point clouds



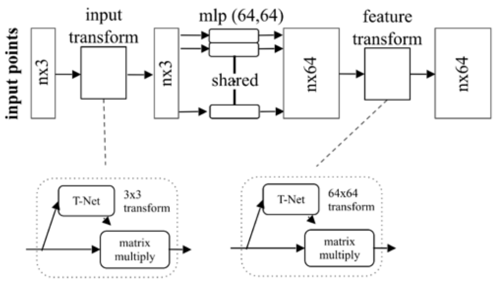
2.2. PointNet classification network

§2. Constructing neural networks on point clouds



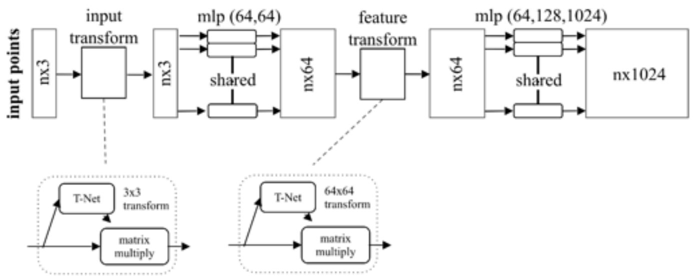
2.2. PointNet classification network

§2. Constructing neural networks on point clouds



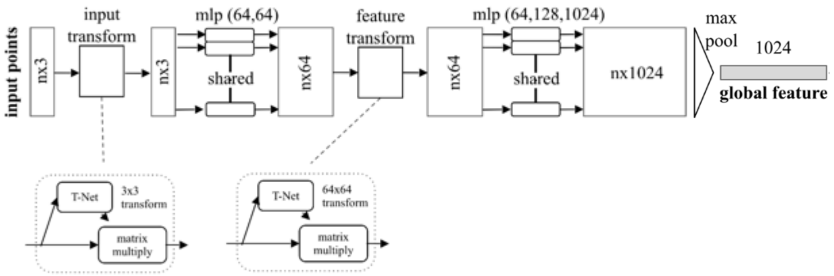
2.2. PointNet classification network

§2. Constructing neural networks on point clouds



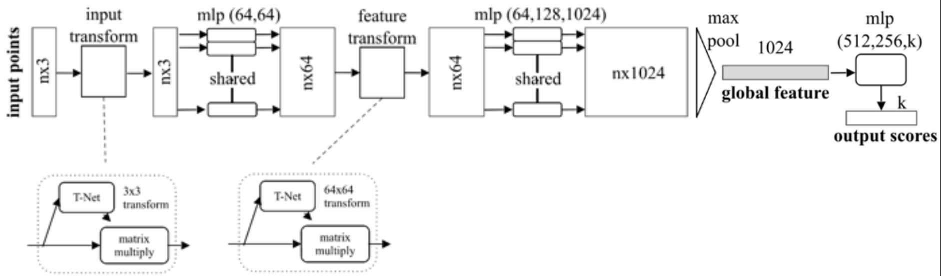
2.2. PointNet classification network

§2. Constructing neural networks on point clouds



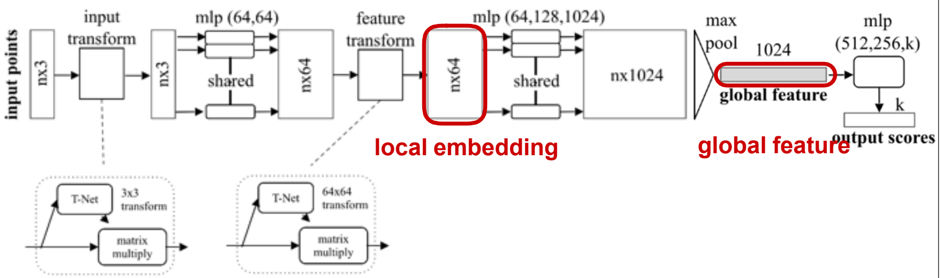
2.2. PointNet classification network

§2. Constructing neural networks on point clouds



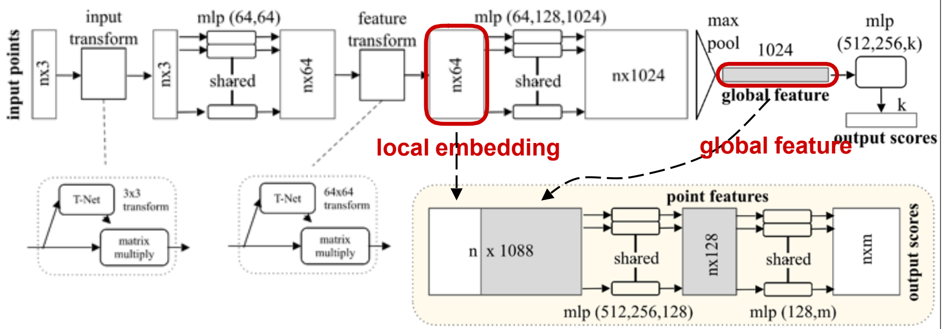
2.2. PointNet segmentation network

§2. Constructing neural networks on point clouds



2.2. PointNet segmentation network

§2. Constructing neural networks on point clouds

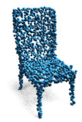


Learning with point clouds: tasks and results

2.3. Tasks: object classification

§2. Constructing neural networks on point clouds

→ "Airplane"



→ "Chair"

SHAPENET

2.3. Results: object classification

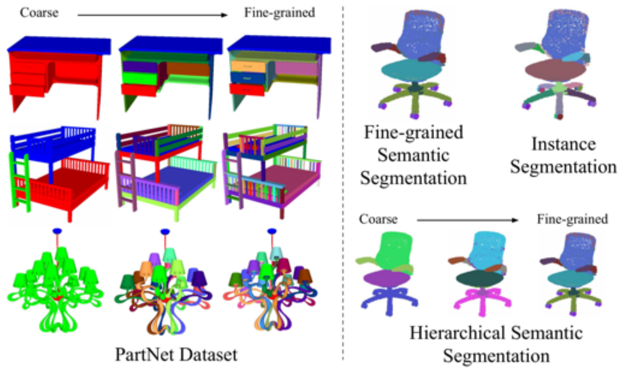
§2. Constructing neural networks on point clouds

		input	#views	accuracy avg. class	accuracy overall
	SPH [12]	mesh	-	68.2	-
3D CNNs	3DShapeNets [29]	volume	1	77.3	84.7
	VoxNet [18]	volume	12	83.0	85.9
	Subvolume [19]	volume	20	86.0	**89.2**
	LFD [29]	image	10	75.5	-
	MVCNN [24]	image	80	**90.1**	-
	Ours baseline	point	-	72.6	77.4
	Ours PointNet	point	1	86.2	**89.2**

Dataset: ModelNet40; metric: 40-class classification accuracy (%)

2.3. Tasks: object classification

§2. Constructing neural networks on point clouds

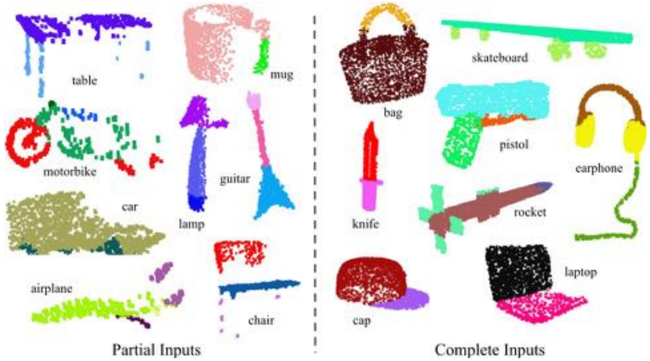


Mo, K., Zhu, S., Chang, A. X., Yi, L., Tripathi, S., Guibas, L. J., & Su, H. (2019). Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 909-918).

2.3. Results: object part segmentation

Skoltech

§2. Constructing neural networks on point clouds



ShapeNetPart benchmark

2.3. Results: object part segmentation

Skoltech

§2. Constructing neural networks on point clouds

	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 [28]	-	63.2	-	-	-	73.5	-	-	-	74.4	-	-	-	-	-	-	74.8
Yi [30]	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**

dataset: ShapeNetPart; metric: mean IoU (%)

2.3. Results: semantic scene parsing

Skoltech

§2. Constructing neural networks on point clouds

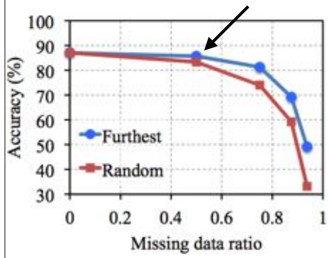
dataset: Stanford 2D-3D-S (Matterport scans)

2.3. Results: robustness to data corruption

Skoltech

§2. Constructing neural networks on point clouds

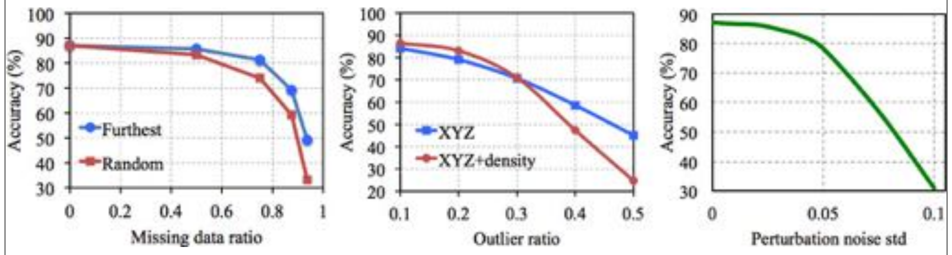
Less than 2% accuracy drop with 50% missing data



dataset: ModelNet40; metric: 40-class classification accuracy (%)

2.3. Results: robustness to data corruption

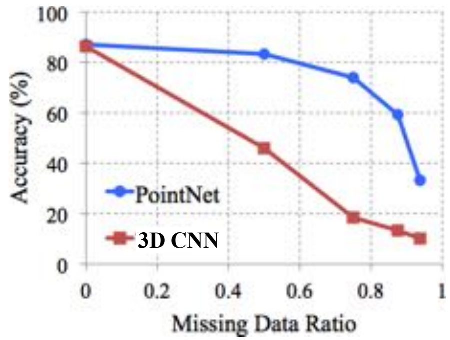
§2. Constructing neural networks on point clouds



dataset: ModelNet40; metric: 40-class classification accuracy (%)

2.3. Results: robustness to data corruption

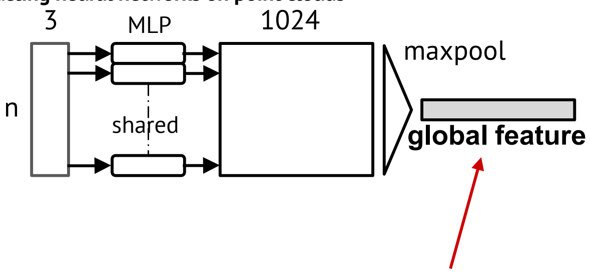
§2. Constructing neural networks on point clouds



Why is PointNet so robust to missing data?

2.3. Results: visualizing global point cloud features

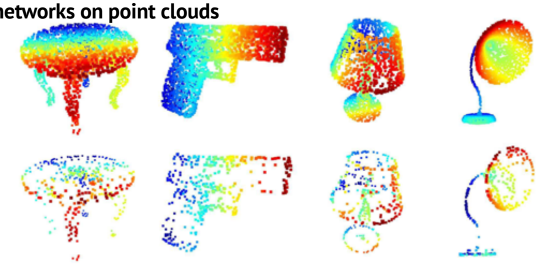
§2. Constructing neural networks on point clouds



Which input points are contributing to the global feature?
(critical points)

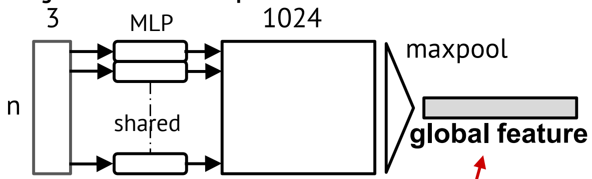
2.3. Results: visualizing global point cloud features

§2. Constructing neural networks on point clouds



2.3. Results: visualizing global point cloud features

§2. Constructing neural networks on point clouds



Which points won't affect the global feature?

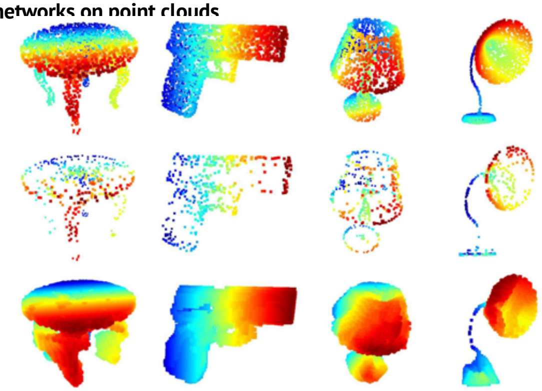
2.3. Results: visualizing global point cloud features

§2. Constructing neural networks on point clouds

Original Shape:

Critical Point Set:

Upper bound set:

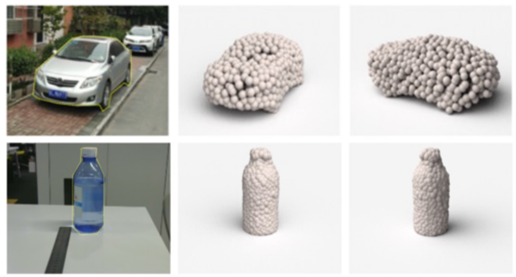


§3. Distances on point sets

3.1. Why distances on point sets?

§3. Distances on point sets

- Point cloud reconstruction tasks:
  - Given an input X (e.g. an image, a corrupted point cloud, …), **produce a complete point cloud**
- Needs to assess quality of point cloud reconstruction
  - Key question: **how to measure differences between point sets?**
  - Common losses: L2



Input Reconstructed 3D point cloud

Figure credit: A Point Set Generation Network for 3D Object Reconstruction from a Single Image

Distances
(and their definition for sets)

3.1. Distances (and their definition for sets)

§3. Distances on point sets

- A function $d(\,\cdot\,) : X \times X \to \mathbb{R}^+$ is a **metric** on a nonempty set X if for all $x, y, z, \in X$:
  1. $d(x, y) = 0$ iff $x = y$.
  2. $d(x, y) = d(y, x)$ (symmetry).
  3. $d(x, z) \leqslant d(x, y) + d(y, z)$ (triangle inequality).
- A function $d(\,\cdot\,) : X \times X \to \mathbb{R}^+$ is a **distance function** on a nonempty set X **if it satisfies 1. and 2.**

3.1. Distances (and their definition for sets)

§3. Distances on point sets

- Extend a distance (or metric) on X to a distance (or metric) on a collection of all non-empty subsets of X
- Define distance from element $x \in X$ to a set Y:

$$d(x, Y) = \inf_{y \in Y} d(x, y)$$

- One obvious choice – **Hausdorff metric:**

$$d_H(X, Y) = \max \left\{ \sup_{x \in X} d(x, Y), \sup_{y \in Y} d(X, y) \right\}$$

  - $d_H(X, Y)$ is a metric if $d(x, y)$ is a metric

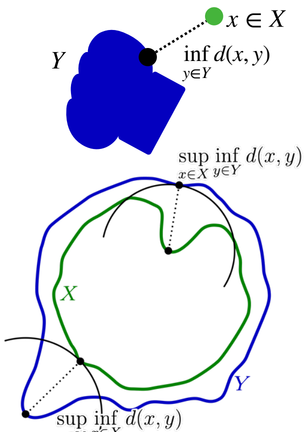


Figure credit: Wikipedia

3.1. Distances (and their definition for sets)

§3. Distances on point sets

- Assume $d(x, y)$ is a given metric
- Define a correspondence $x \to \phi_x(x) \in Y, y \to \phi_t(y) \in X$
  - E.g. $\phi_x(x) = \arg\min_{y \in Y} d(x, y)$
- Pick an aggregation operator $\mathrm{Agg}_{x \in X}$ and a function $\gamma(a, b)$
- Often distance measures on sets take the general form

$$d_{\text{set}}(X, Y) = \gamma\left(\mathrm{Agg}_{x \in X} d(x, \phi_x(x)), \mathrm{Agg}_{y \in Y} d(y, \phi_y(y))\right)$$

  - E.g. Hausdorff distance: $\gamma = \max, \mathrm{Agg} = \sup$

Examples of commonly employed metrics

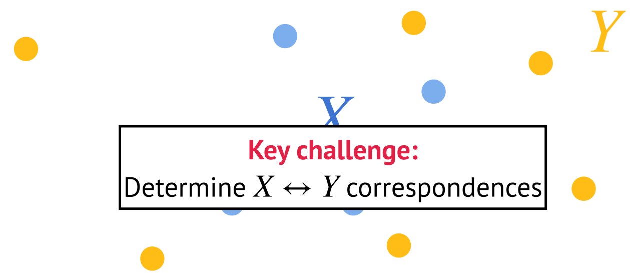
3.2. Desiderata: commonly employed metrics

§3. Distances on point sets

- Differentiable with respect to point locations
- Efficient to compute, as data will be forwarded and back-propagated for many times
- Robust against small number of outlier points in the sets (e.g. Hausdorff distance would fail)

3.2. Constructing metrics on point sets

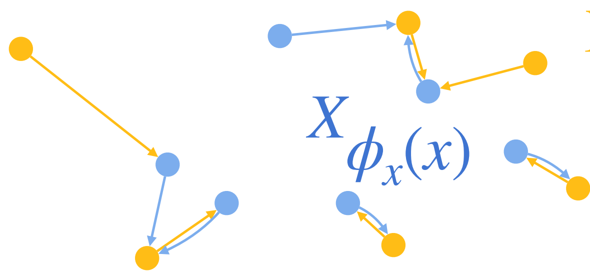
§3. Distances on point sets



- Given two sets X and Y, measure their discrepancy

3.2. Distances on point sets: Chamfer distance

§3. Distances on point sets



$\gamma(a, b) = a + b$

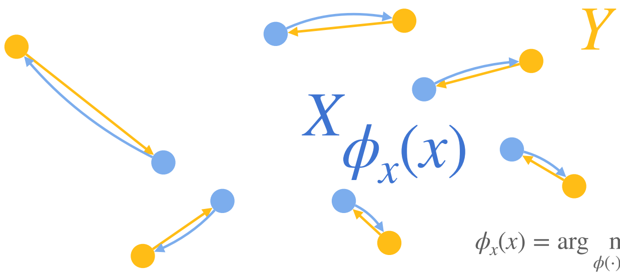
Agg $= \sum$

$\phi_x(x) = \arg\min_{y \in Y} d(x, y)$

Chamfer distance: $d_{\mathrm{CD}}(X, Y) = \sum_{x \in X} \min_{y \in Y} \|x - y\|_2^2 + \sum_{y \in Y} \min_{x \in X} \|x - y\|_2^2$

3.2. Distances on point sets: EMD

§3. Distances on point sets



$|X| = |Y|$

$\gamma(a, b) = a$

$\mathrm{Agg} = \sum$

$\phi_x(x) = \arg \min_{\phi(\cdot):X \to Y} \sum_{x \in X} \|x - \phi(x)\|_2^2$

Earth Mover's distance: $d_{\mathrm{EMD}}(X, Y) = \min_{\phi_x(\cdot):X \to Y} \sum_{x \in X} \|x - \phi_x(x)\|_2^2$

3.2. Summary: commonly employed metrics

§3. Distances on point sets

- Chamfer distance: $d_{\mathrm{CD}}(X, Y) = \sum_{x \in X} \min_{y \in Y} \|x - y\|_2^2 + \sum_{y \in Y} \min_{x \in X} \|x - y\|_2^2$
- Earth Mover's distance: $d_{\mathrm{EMD}}(X, Y) = \min_{\phi_x(\cdot):X \to Y} \sum_{x \in X} \|x - \phi_x(x)\|_2^2$
  - Simple functions of coordinates
  - In general positions, the correspondence is unique
  - With infinitesimal movement, the correspondence does not change
- **Conclusion: differentiable almost everywhere**