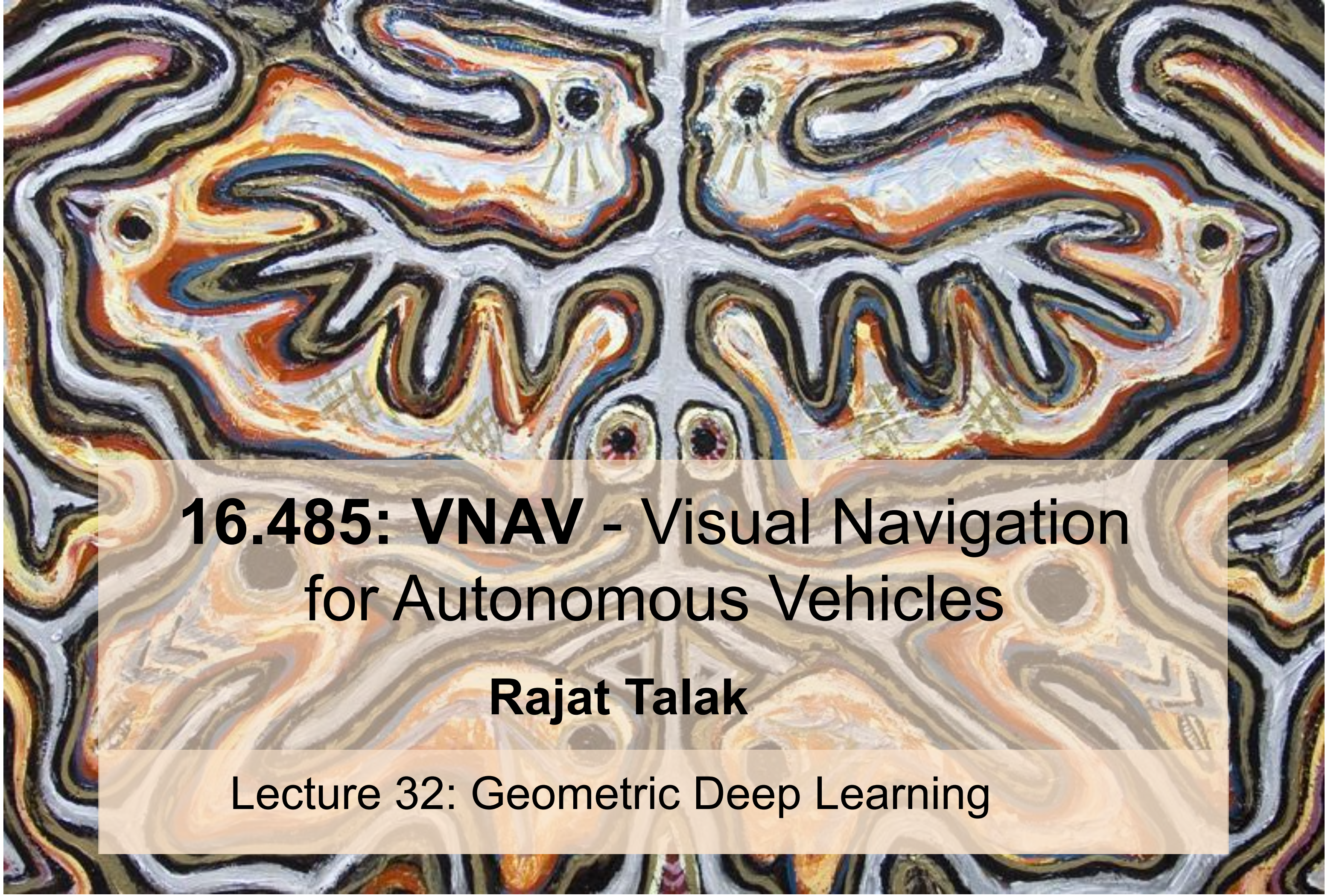


16.485: VNAV - Visual Navigation for Autonomous Vehicles

Rajat Talak

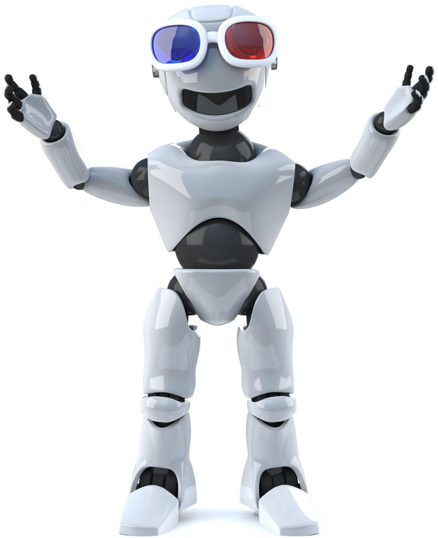
Lecture 32: Geometric Deep Learning

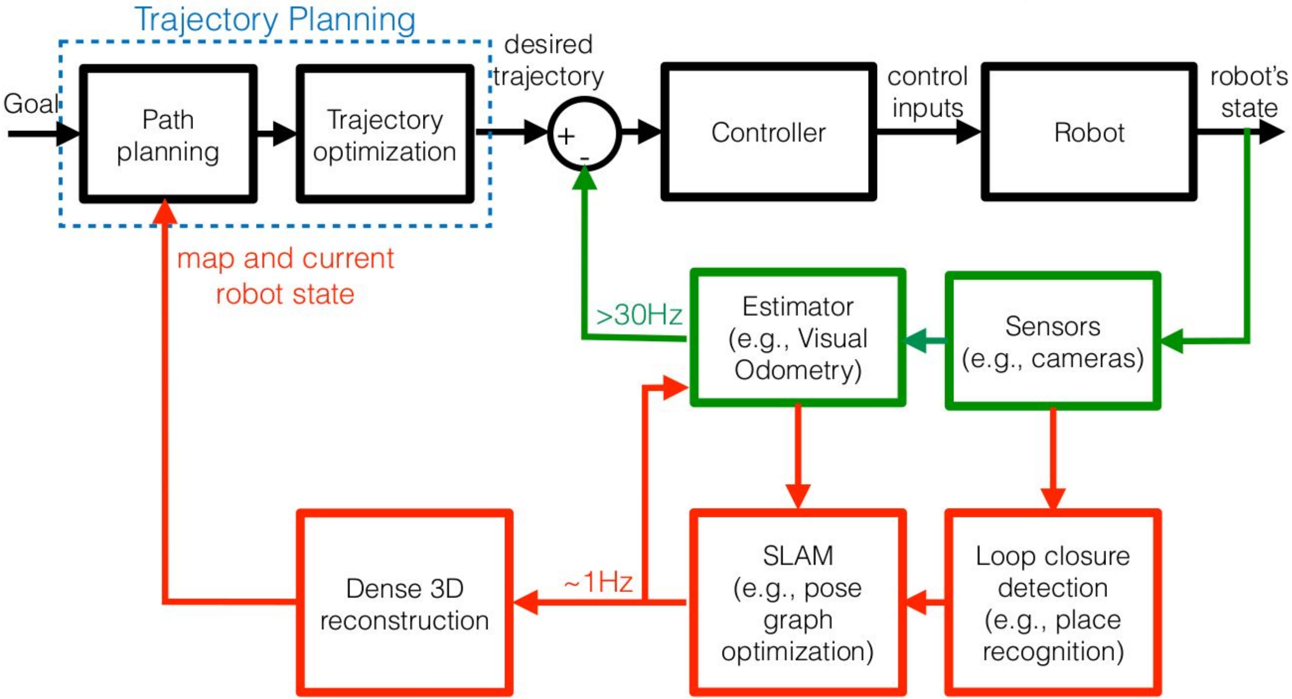
Image: “Nazca Symmetry” Kazuya Akimoto Art Museum

Recap: Motivation

3D Geometric Reconstruction is not enough!

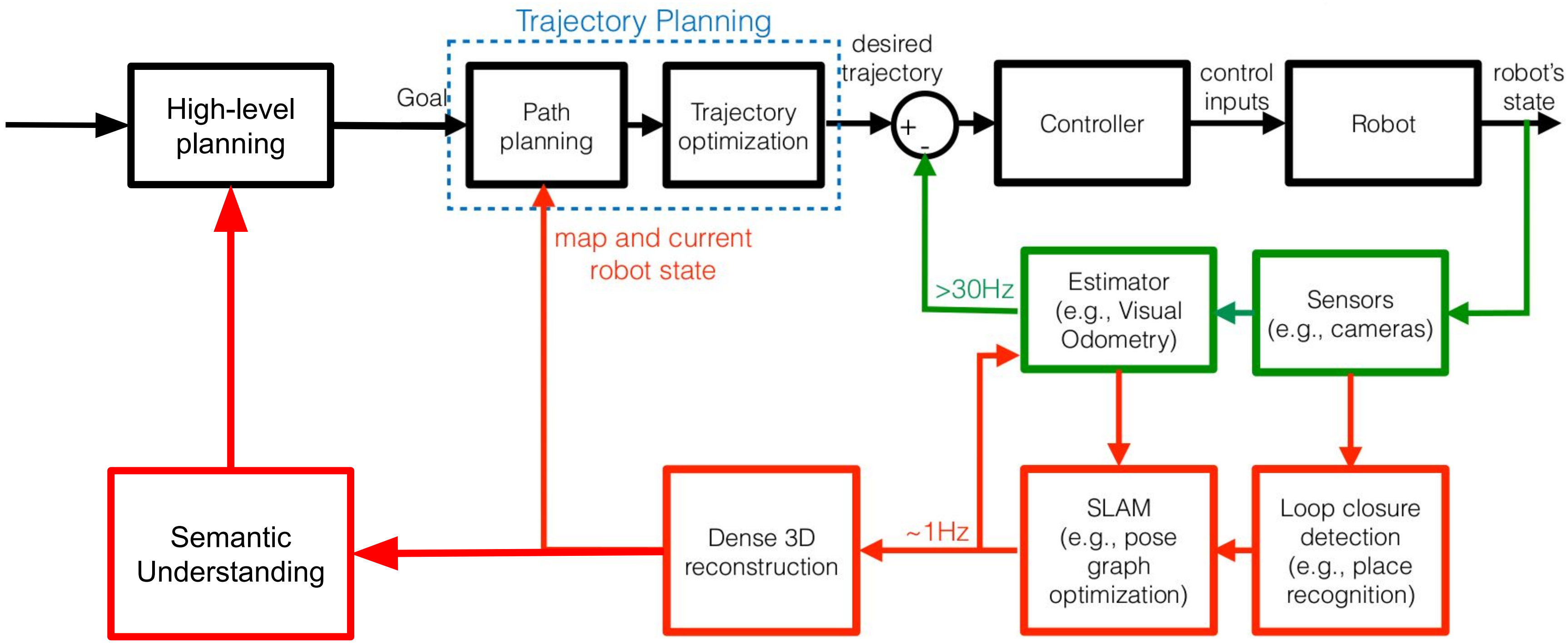
Need for Semantic Understanding





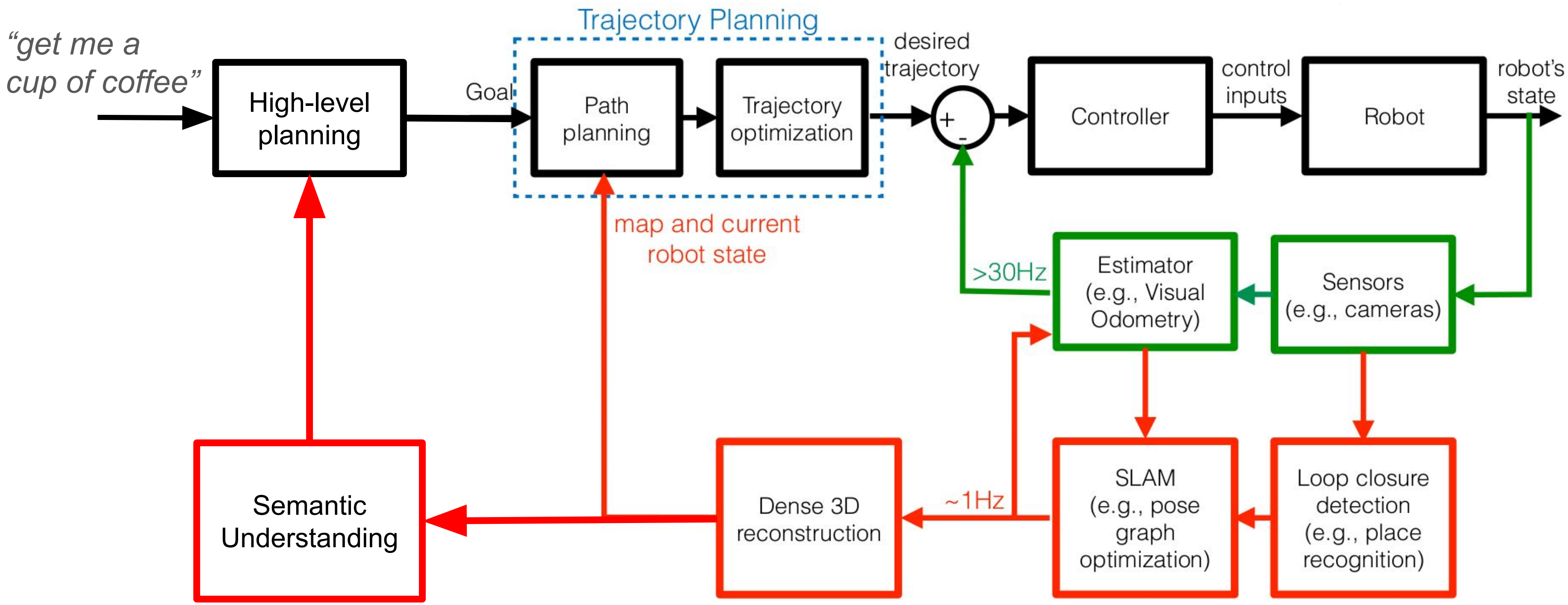
Recap: Motivation

3D Geometric Reconstruction is not enough!



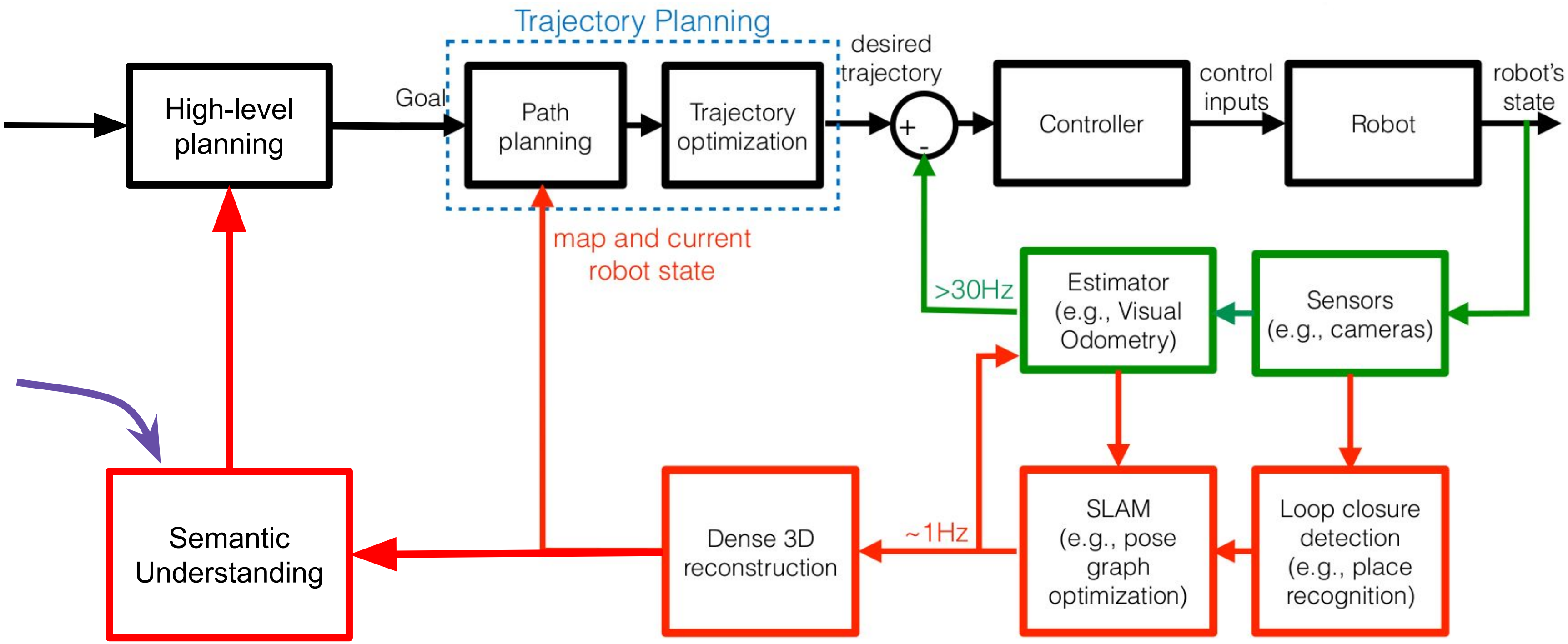
Recap: Motivation

3D Geometric Reconstruction is not enough!



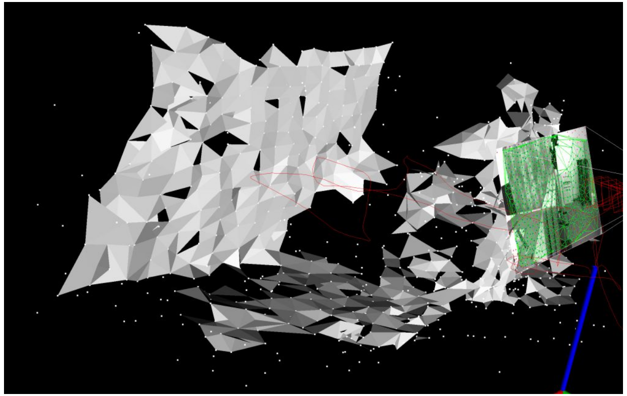
Recap: Motivation

3D Geometric Reconstruction is not enough!

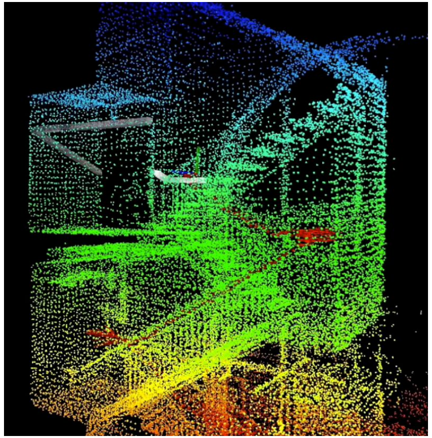


Semantic Understanding

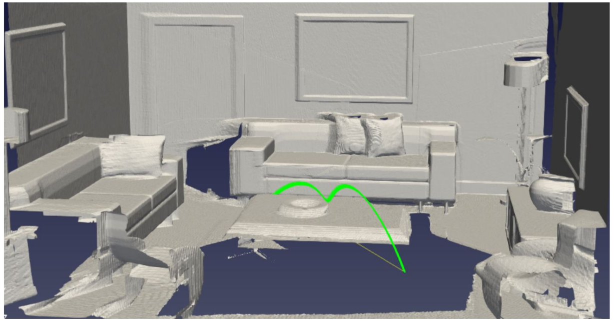
Need for Deep Learning Architectures on Point Clouds, Voxels, Meshes, Graphs



Mesh

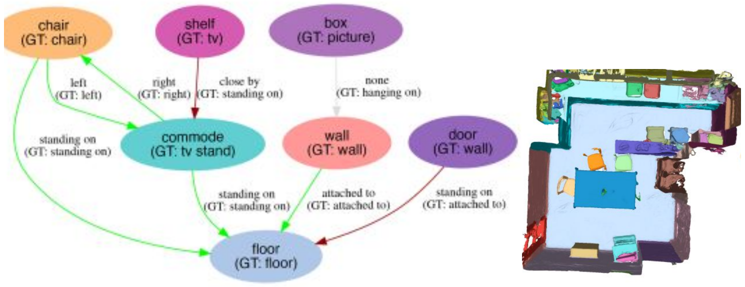


Point Cloud



Voxel

Vespa et al. “Efficient Octree-based Volumetric SLAM Supporting Signed-Distance and Occupancy Mapping” RAL 2017



Graph

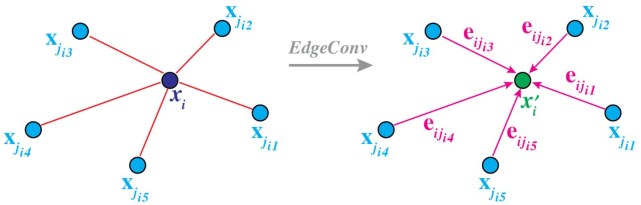
Wald et al. “Learning 3D Semantic Scene Graphs from 3D Indoor Reconstruction” 2020

Recap: Architectures Discussed

Voxel

VoxNet

OctNet



Point Cloud

PointNet

PointNet++

EdgeConv

KPConv

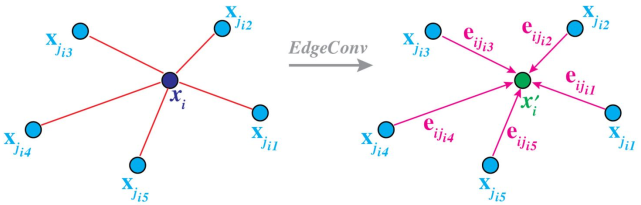
Point Transformer

Today! For the first part …

Voxel

VoxNet

OctNet



Point Cloud

PointNet

PointNet++

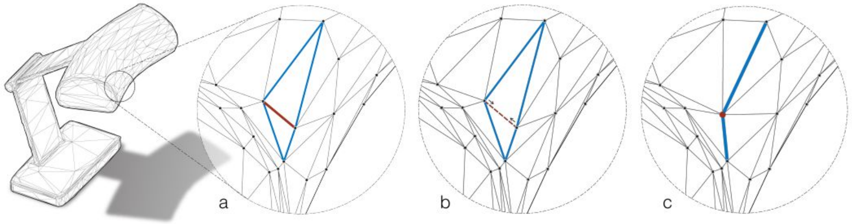
EdgeConv

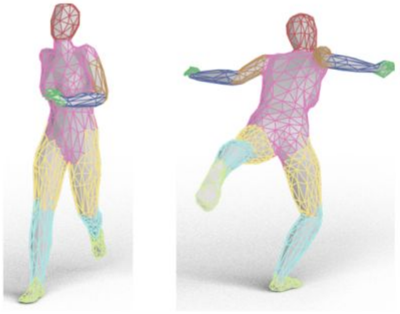
KPConv

Point Transformer

Mesh

MeshCNN





Point Cloud-based Architectures

Efficient than voxel based architectures

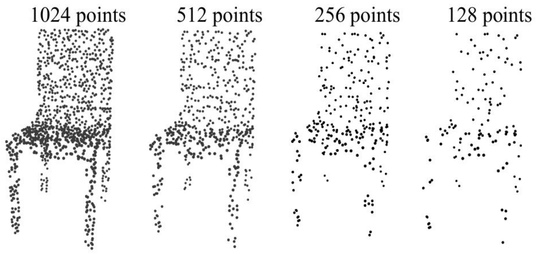
Suitable for point cloud inputs (LiDAR or RGB-D)

Point Cloud-based Architectures

Efficient than voxel based architectures

Suitable for point cloud inputs (LiDAR or RGB-D)

Point clouds may not be the best way to represent 3D shapes

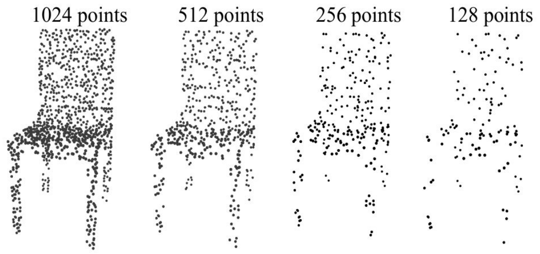


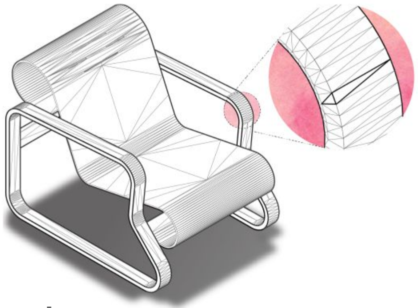
Point Cloud-based Architectures

Efficient than voxel based architectures

Suitable for point cloud inputs (LiDAR or RGB-D)

Point clouds may not be the best way to represent 3D shapes



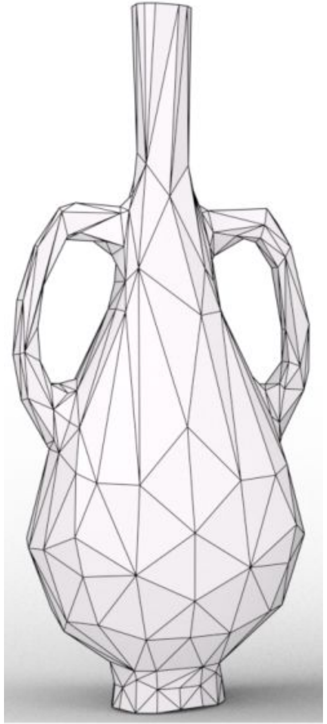


Mesh

Mesh

Mesh Representation

Mesh = Vertices, Faces, Edges



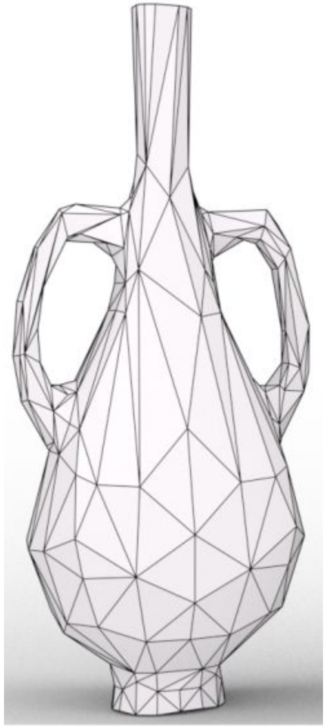
source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

Mesh Representation

Mesh = Vertices, Faces, Edges

3d locations

$$v = (x, y, z)$$



source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

Mesh Representation

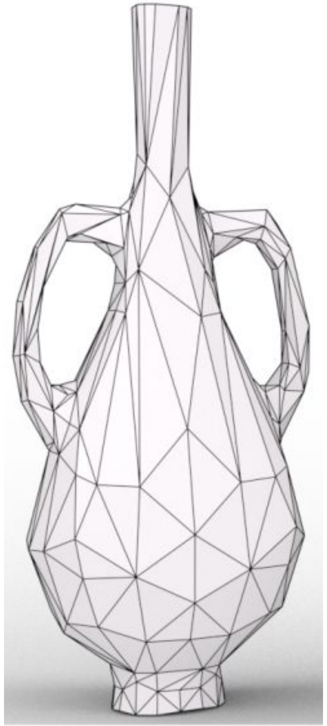
Mesh = Vertices, Faces, Edges

3d locations

$$v = (x, y, z)$$

Triplet of vertices

$$f = (v_1, v_2, v_3)$$



source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

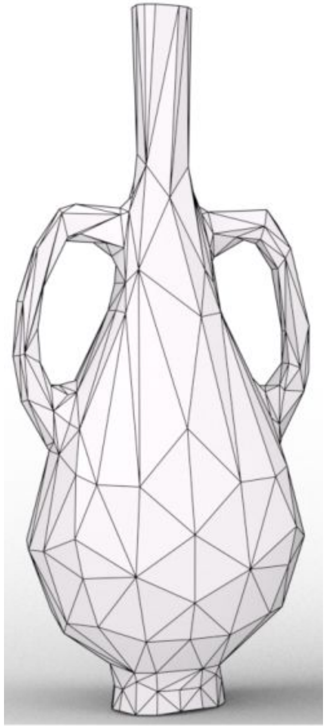
Mesh Representation

Mesh = Vertices, Faces, Edges

3d locations
$$v = (x, y, z)$$

Triplet of vertices
$$f = (v_1, v_2, v_3)$$

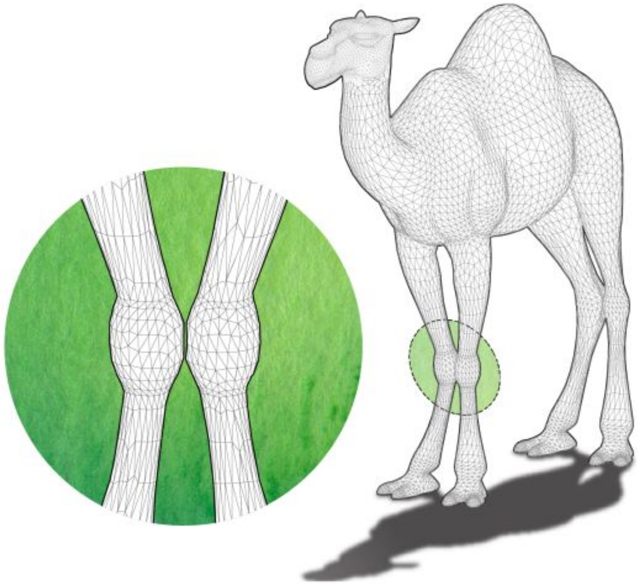
Pair of vertices
$$e = (v_1, v_2)$$



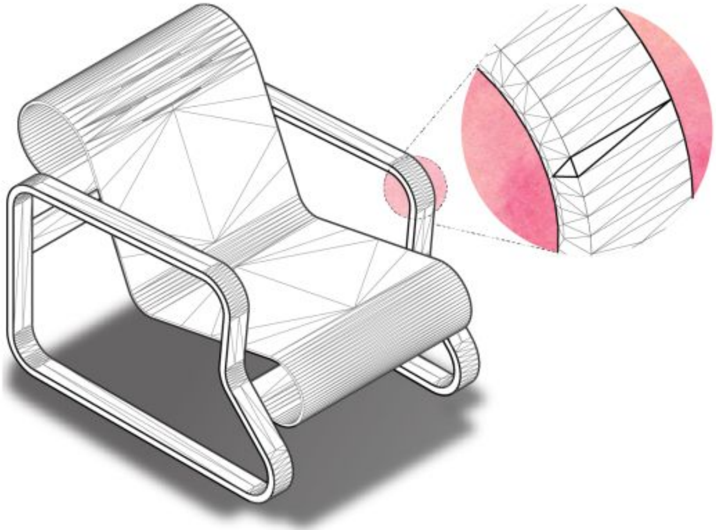
source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

Mesh Connectivity and Local Shape

Conveys distinctness of local shape



Adaptive to non-uniform shape



source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

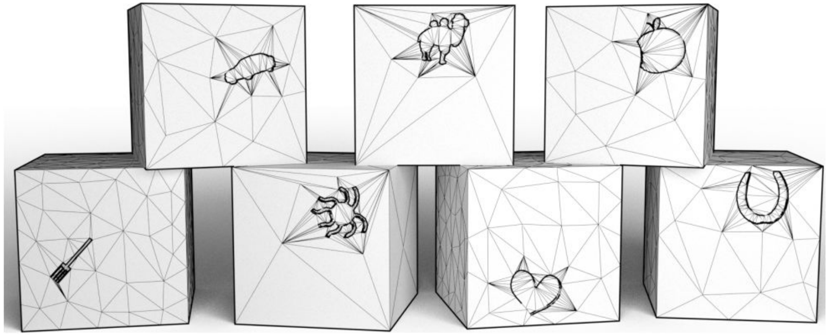
Learning on Meshes

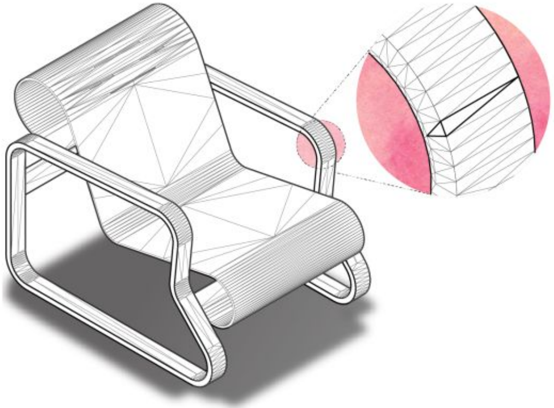
Architectures should be able to exploit this property

Learning on Meshes

Architectures should be able to exploit this property

Problem: non-uniformity of the mesh





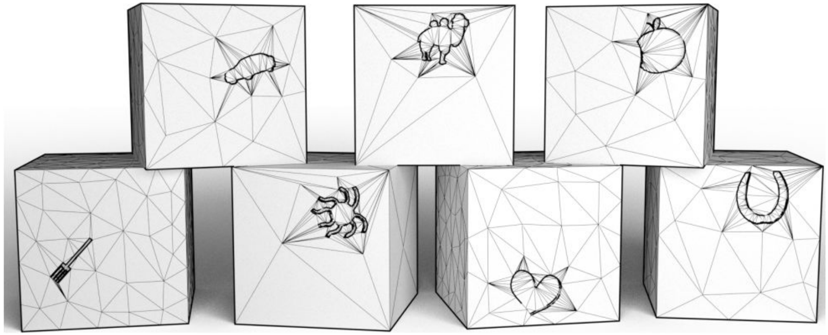
source: Hanocka et al. "MeshCNN: A Network with an Edge" ACM Trans. Graph. 2019

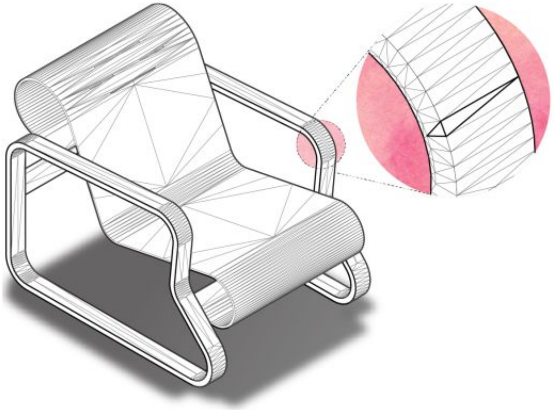
Learning on Meshes

Architectures should be able to exploit this property

Each vertex has varying number of neighbors

Problem: non-uniformity of the mesh



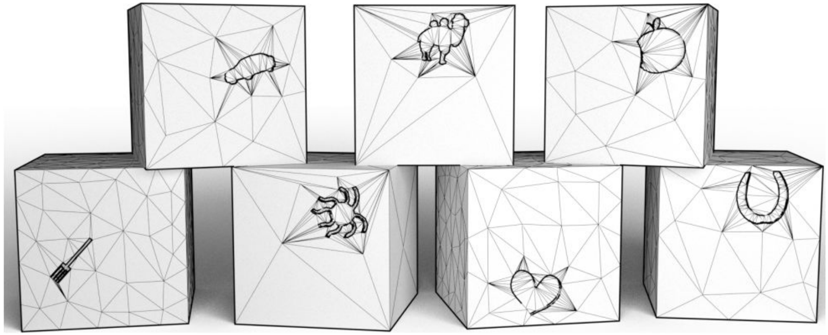


source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

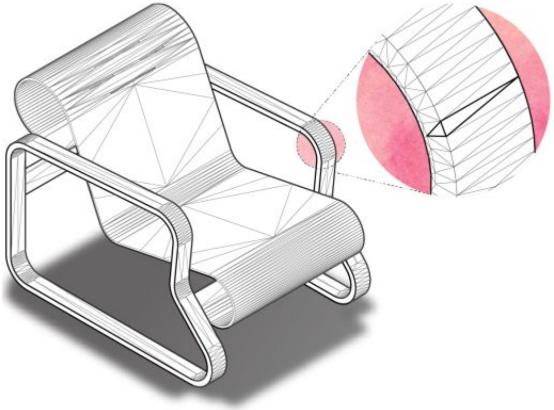
Learning on Meshes

How do we define convolution, pooling, and unpooling on this?

Problem: non-uniformity of the mesh



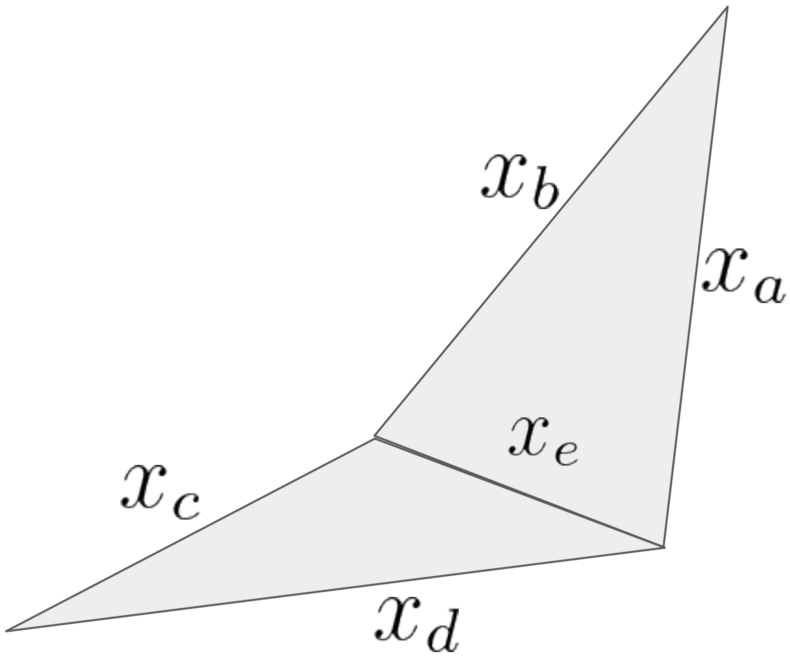
Each vertex has varying number of neighbors



source: Hanocka et al. "MeshCNN: A Network with an Edge" ACM Trans. Graph. 2019

MeshCNN

Every edge has two adjacent faces
and four adjacent edges

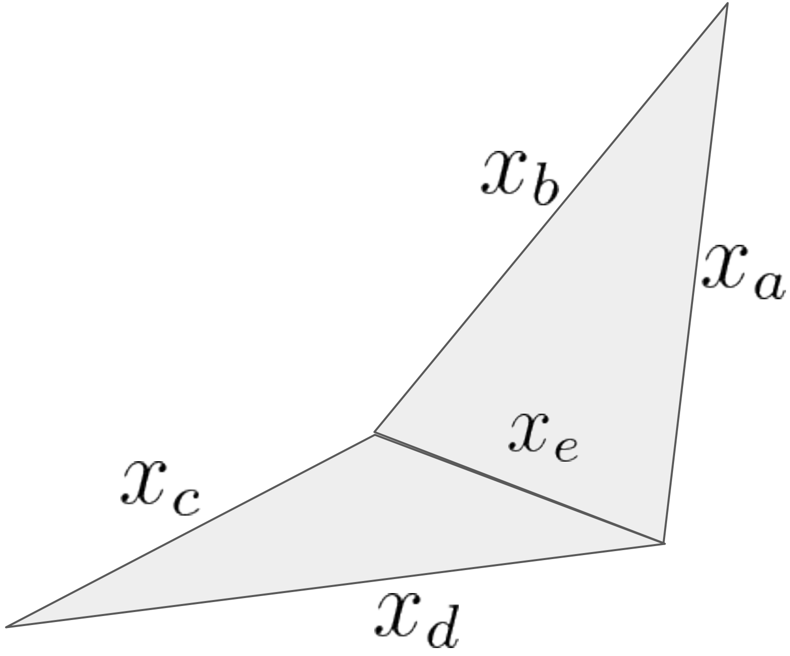


MeshCNN

Every edge has two adjacent faces and four adjacent edges

Operates over mesh edges

Generates and updates representation vectors over mesh edges



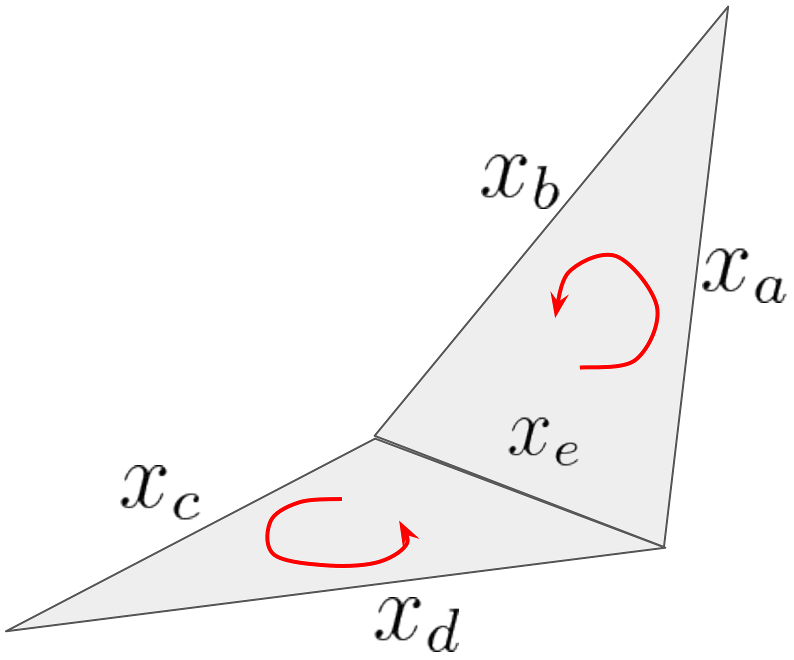
Ordering Neighboring Edges

Always order counter-clockwise

Two possibilities

$$(x_a, x_b, x_c, x_d)$$

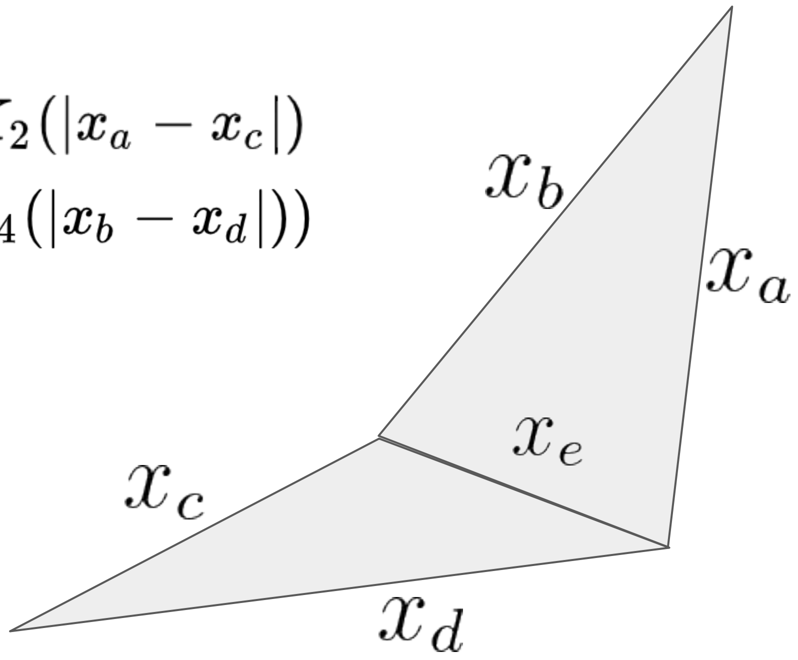
$$(x_c, x_d, x_a, x_b)$$



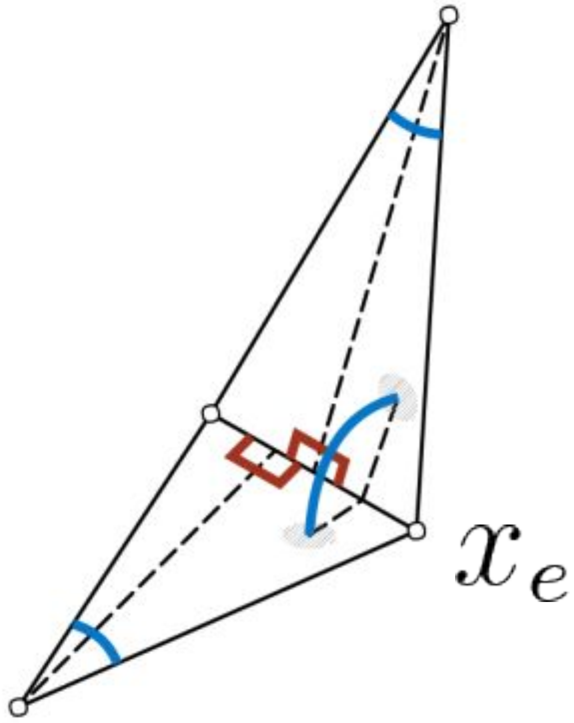
Aggregation should be invariant to these two possibilities

Updating Edge Features

$$x'_e = \sigma\left(Kx_e + K_1(x_a + x_c) + K_2(|x_a - x_c|) + K_3(x_b + x_d) + K_4(|x_b - x_d|)\right)$$



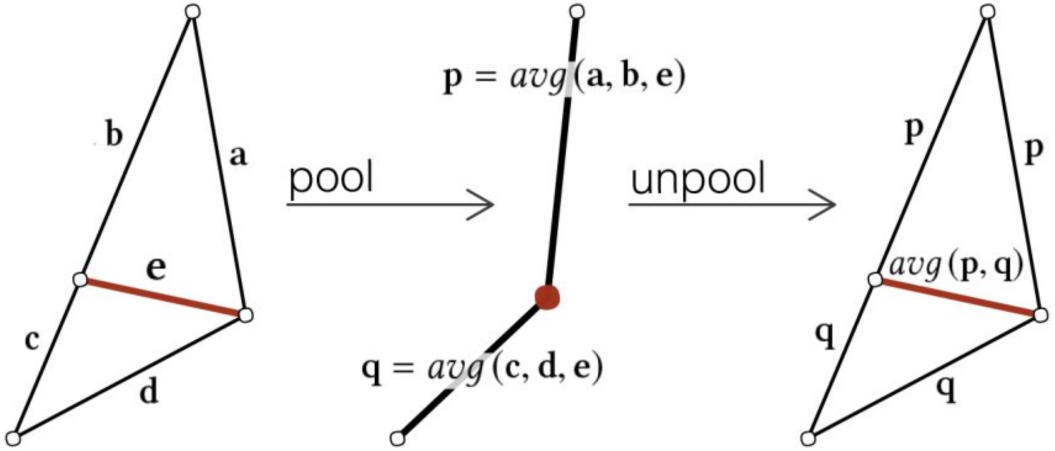
Initial features



Vector of length 5

Invariant to translation, rotation, and uniform scale

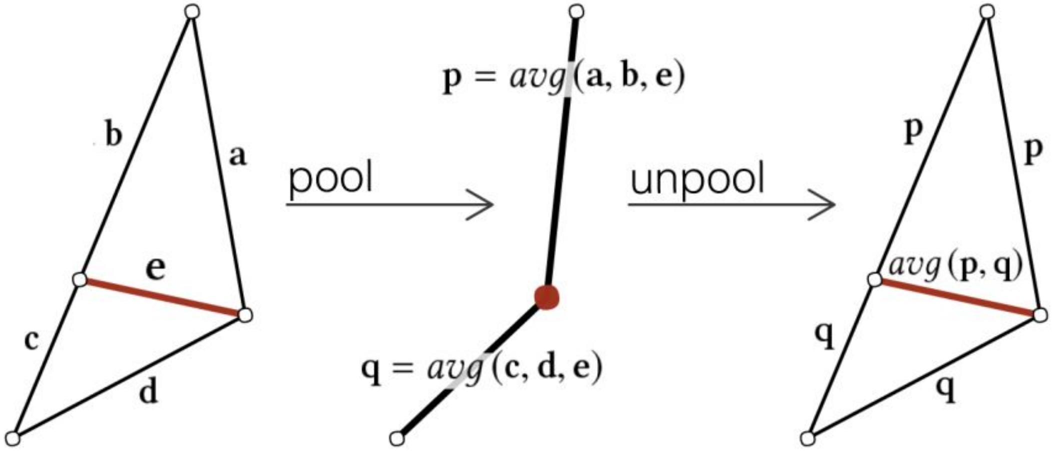
Pooling and Unpooling



In the figure a is $\boldsymbol{x}_a$...

source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

Pooling and Unpooling

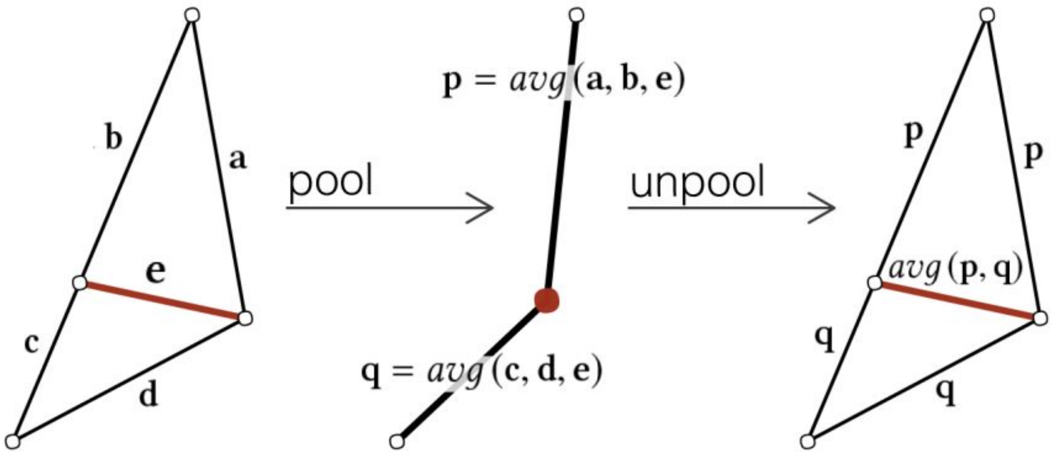


edges with N smallest feature vector are collapsed at each layer

In the figure a is $\boldsymbol{x}_a$...

source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

Pooling and Unpooling



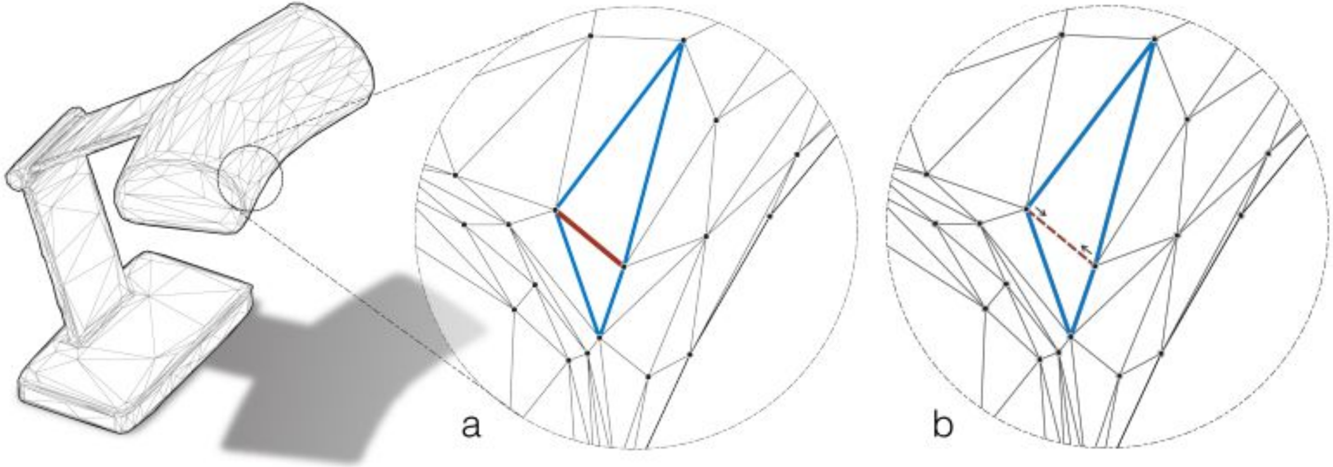
edges with N smallest feature vector are collapsed at each layer

$\boxed{\text{in L2 norm } ||e||_2}$

In the figure a is $\boldsymbol{x}_a$...

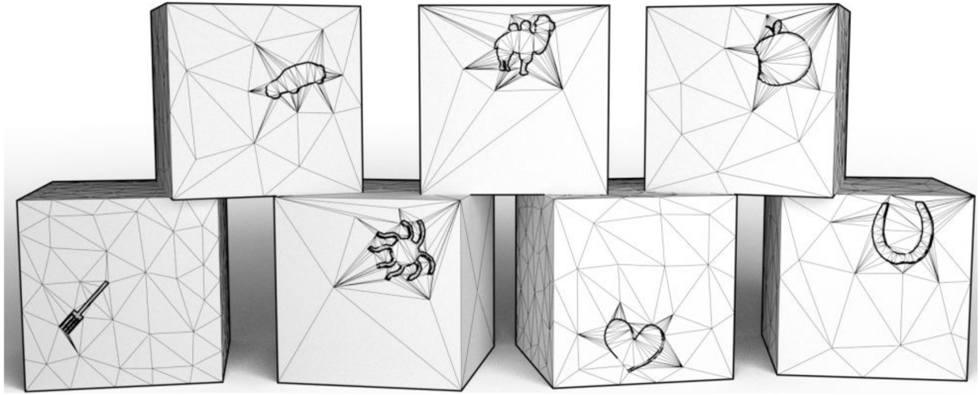
source: Hanocka et al. "MeshCNN: A Network with an Edge" ACM Trans. Graph. 2019

Pooling and Unpooling



source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

MeshCNN: Interesting Results



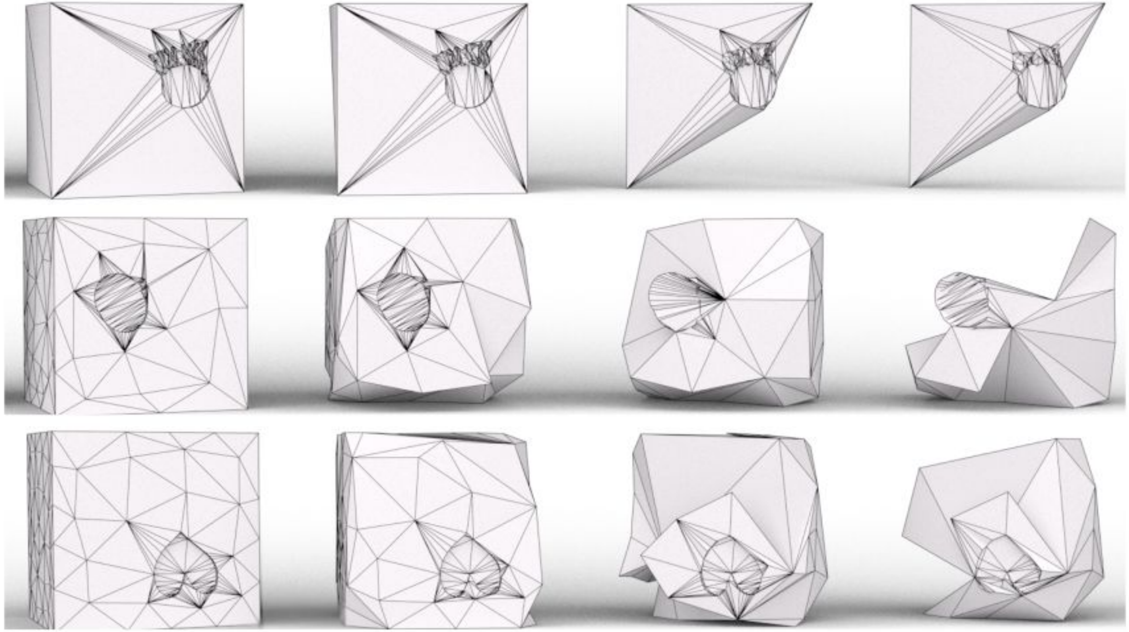
Classifying fine engraved cubes

Cube Engraving Classification

method	input res	test acc
MeshCNN	750	**92.16**%
PointNet++	4096	64.26%

source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

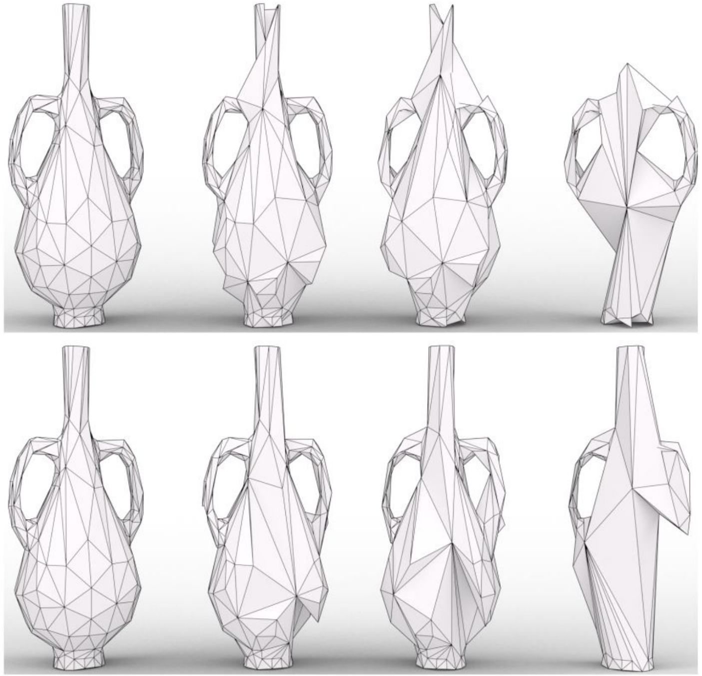
MeshCNN: Interesting Results



preserves important edges required for the task

source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

MeshCNN: Interesting Results



depth

Task 1: Vaze has a handle?

Task 2: Vaze has a neck?

source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

MeshCNN: Human Shape Segmentation

Human Body Segmentation

Method	# Features	Accuracy
MeshCNN	5	**92.30%**
SNGC	3	91.02%
Toric Cover	26	88.00%
PointNet++	3	90.77%
DynGraphCNN	3	89.72%
GCNN	64	86.40%
MDGCNN	64	89.47%

$\left.\right\}$ [2018]

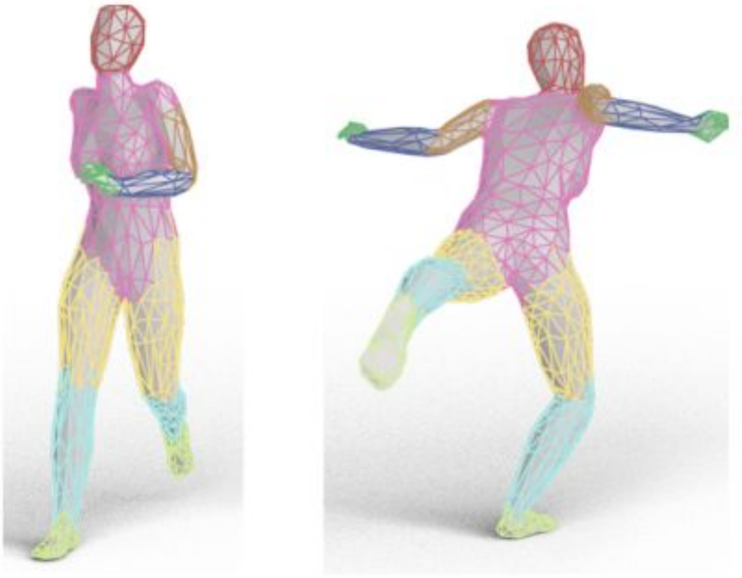
source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

Mesh based Architectures

More structure. Opportunity for the architecture to be more expressive.

Computationally expensive than Point Cloud based architectures.

- Pooling, unpooling, manifoldness

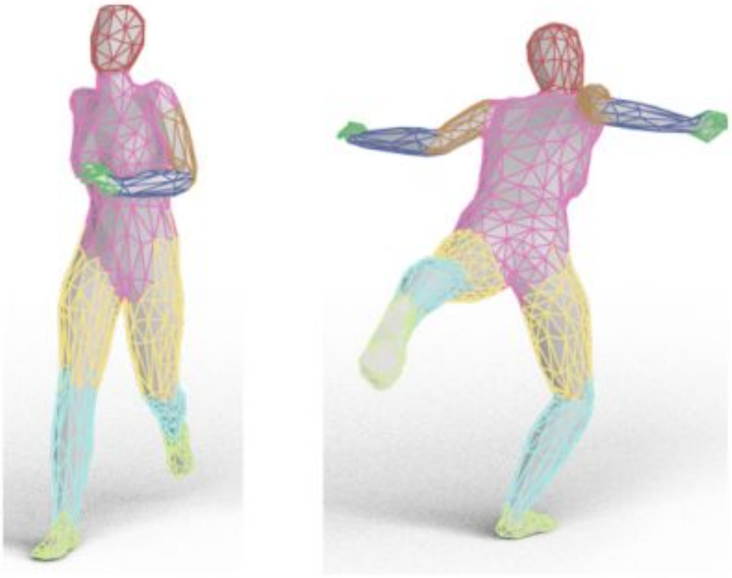


source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

Mesh based Architectures

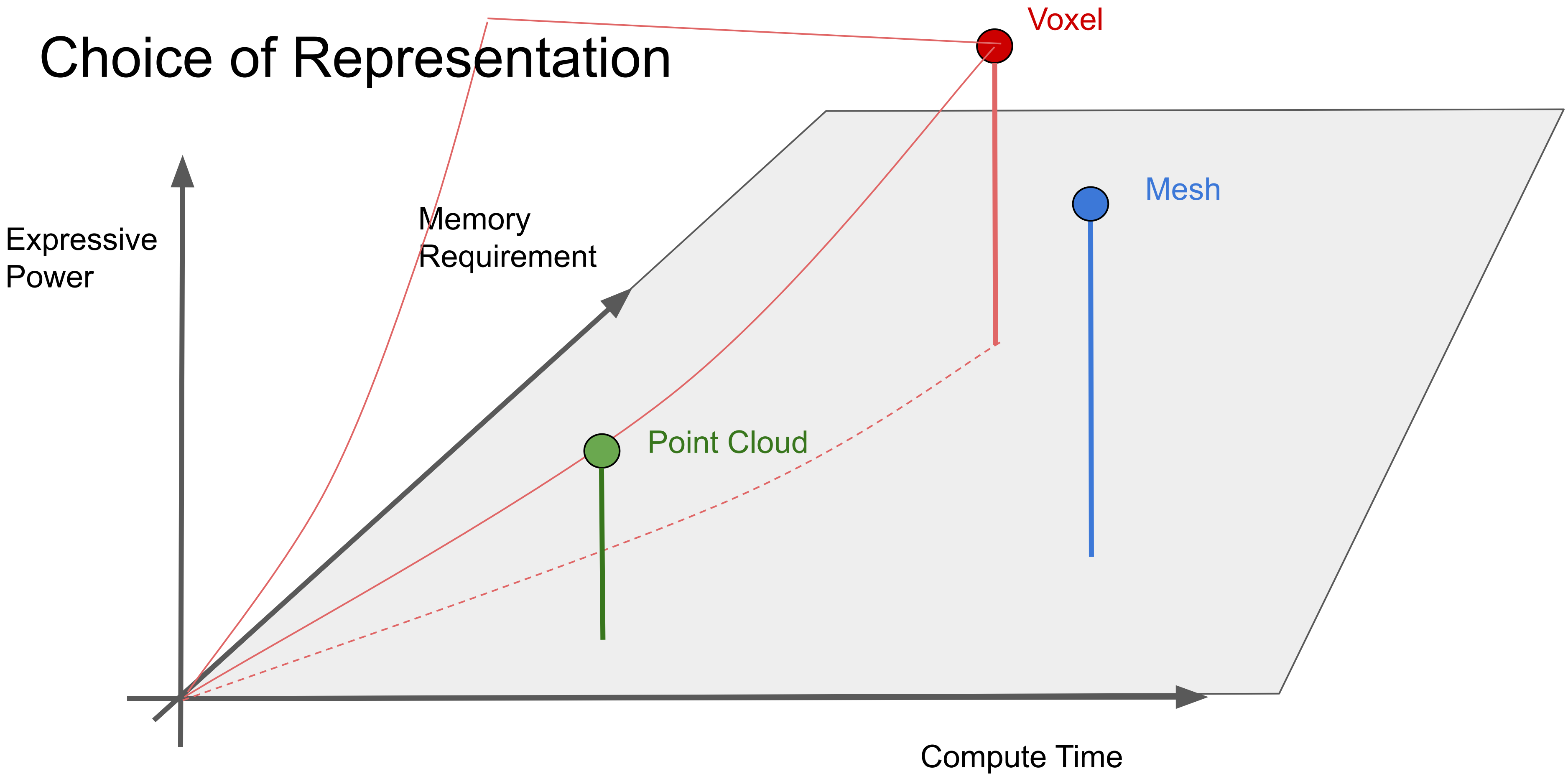
Different meshes can represent the same thing

Data Augmentation

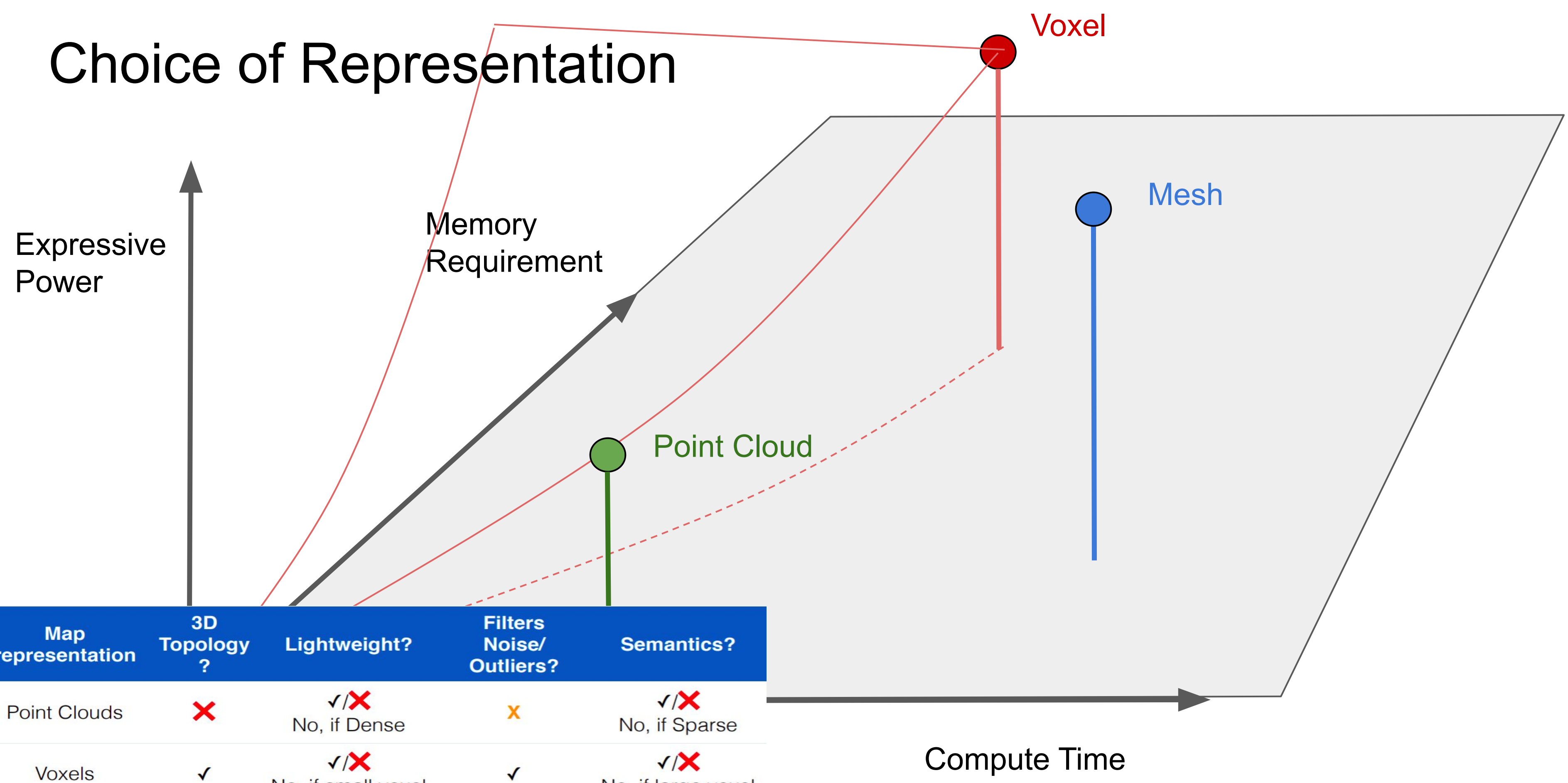


source: Hanocka et al. “MeshCNN: A Network with an Edge” ACM Trans. Graph. 2019

Choice of Representation

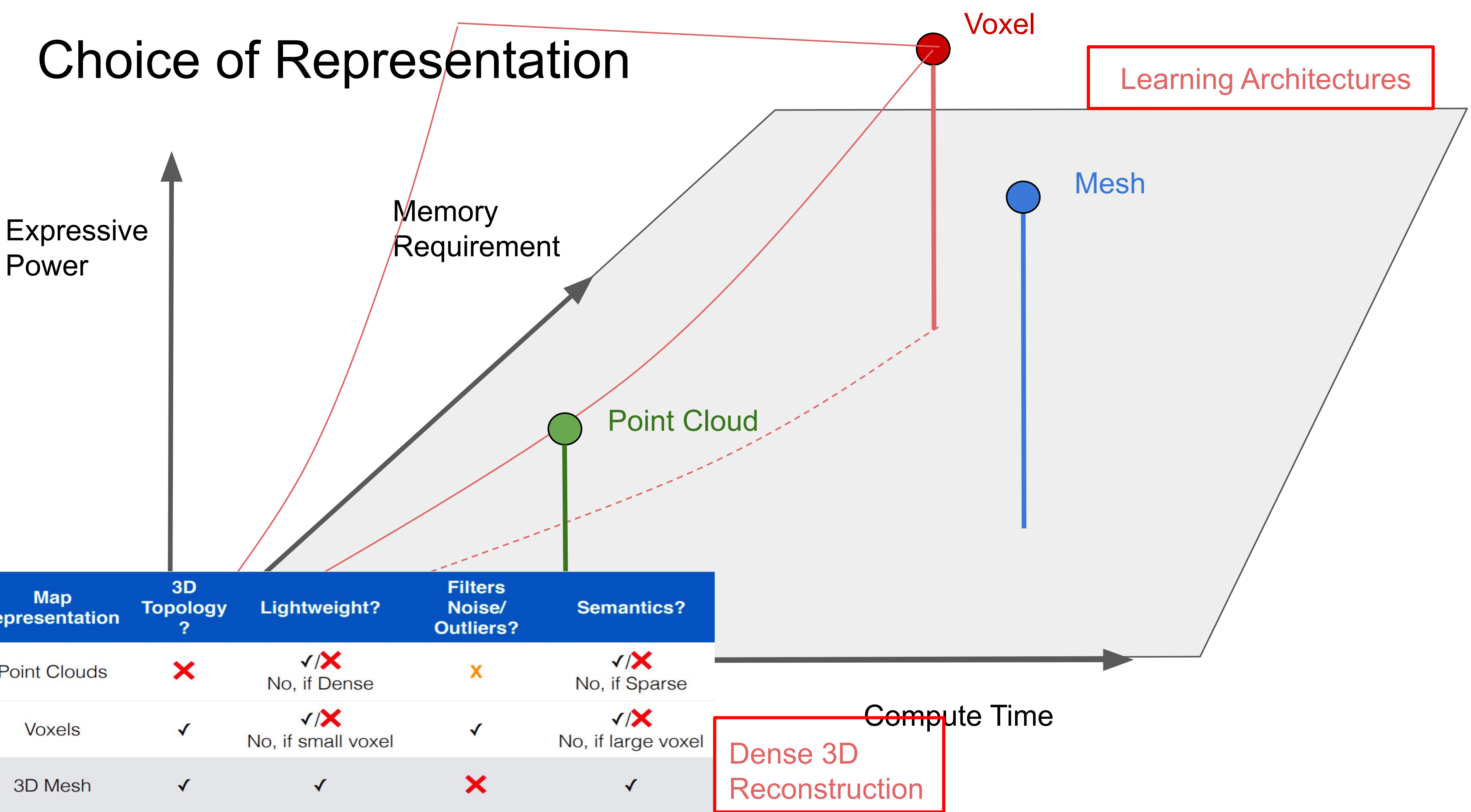


Choice of Representation



Map representation	3D Topology ?	Lightweight?	Filters Noise/ Outliers?	Semantics?
Point Clouds	✗	✓/✗ No, if Dense	x	✓/✗ No, if Sparse
Voxels	✓	✓/✗ No, if small voxel	✓	✓/✗ No, if large voxel
3D Mesh	✓	✓	✗	✓

Choice of Representation



Map representation	3D Topology ?	Lightweight?	Filters Noise/ Outliers?	Semantics?
Point Clouds	✗	✓/✗ No, if Dense	x	✓/✗ No, if Sparse
Voxels	✓	✓/✗ No, if small voxel	✓	✓/✗ No, if large voxel
3D Mesh	✓	✓	✗	✓

Second Part

2

Geometric Deep Learning

- Unifying view of developing architectures on all data
- Symmetry
- Equivariance, Invariance, Convolutions
- Unified Blueprint

Provide a unifying framework for developing deep learning architectures

References:
Bronstein et al. “Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges” 2021.
Bronstein et al. “Geometric Deep Learning” Lectures for AMMI, 2021.

Domains and Architectures

Image	2D Grid	CNN	Voxel	3D Grid	VoxNet, OctNet
Point Cloud	Sets	PointNet ...	Time	1D Grid	RNN, LSTM
Mesh, Manifold	Graph	MeshCNN	Graph	Graph	EdgeConv, MeshCNN, GNN ...

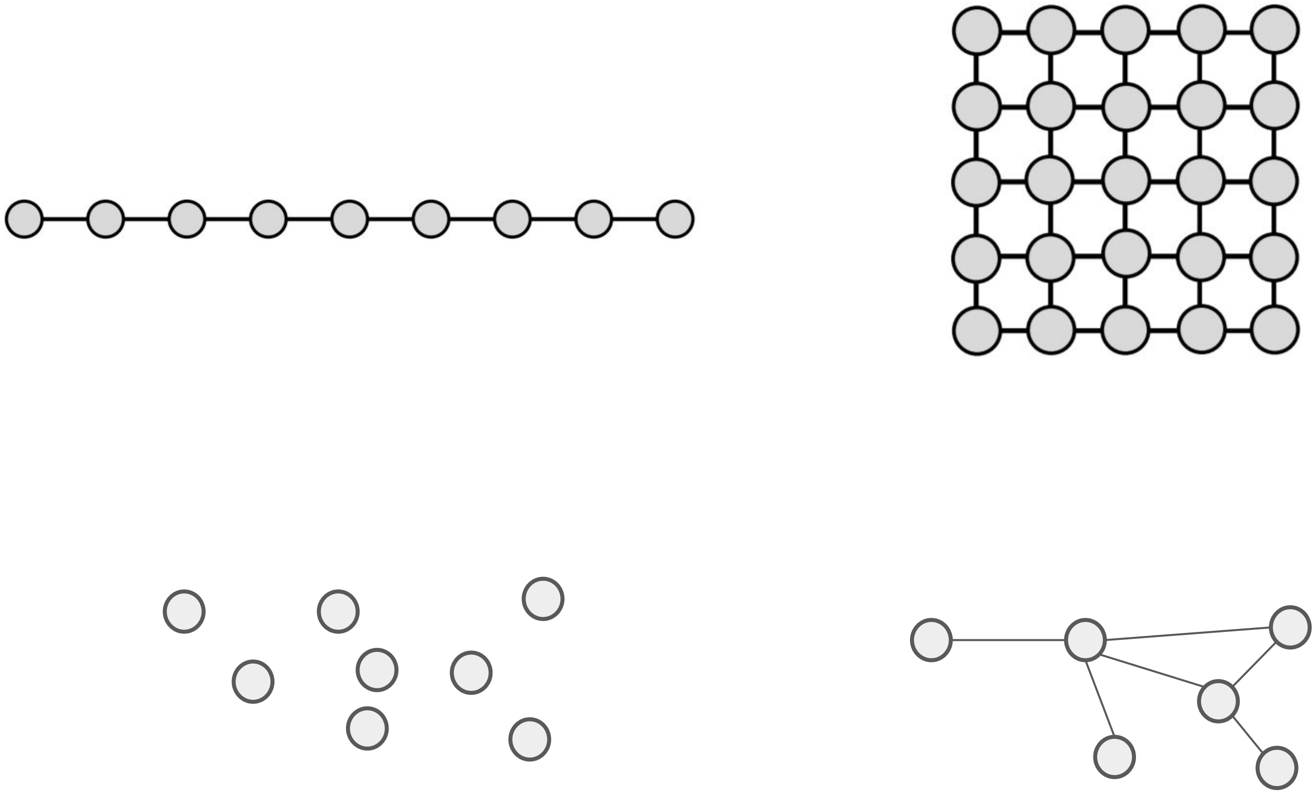
Domains and Architectures

Image	2D Grid	CNN	Voxel	3D Grid	VoxNet, OctNet
Point Cloud	Sets	PointNet ...	Time	1D Grid	RNN, LSTM
Mesh, Manifold	Graph	MeshCNN	Graph	Graph	EdgeConv, MeshCNN, GNN ...

Need an abstraction

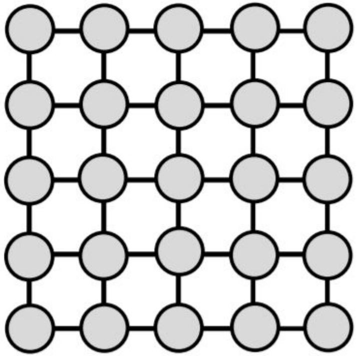
Domain

Ω

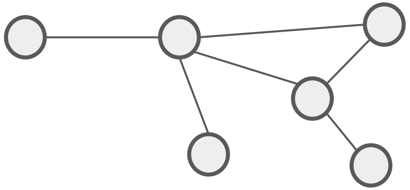


Signals on the Domain

Ω



$$\mathcal{X}(\Omega) = \{x : \Omega \to \mathbb{R}\}$$



Functions

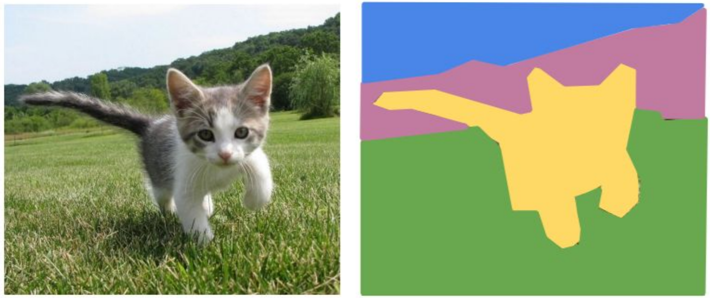
Ω

$$\mathcal{F}_C = \{f : \mathcal{X}(\Omega) \to \mathbb{R}\}$$

$$\mathcal{F}_S = \{f : \mathcal{X}(\Omega) \to \mathcal{X}(\Omega)\}$$

Functions

Ω

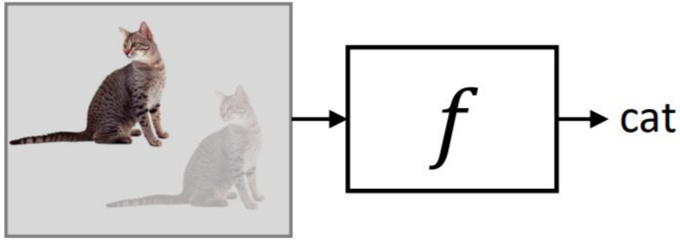


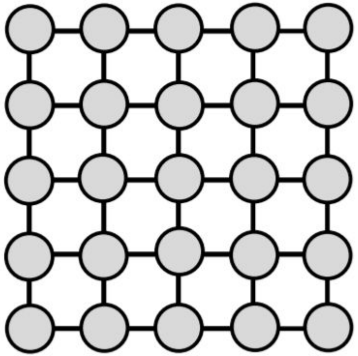
$$\mathcal{F}_C = \{f : \mathcal{X}(\Omega) \to \mathbb{R}\}$$ Classification

$$\mathcal{F}_S = \{f : \mathcal{X}(\Omega) \to \mathcal{X}(\Omega)\}$$ Segmentation

Symmetries

Ω





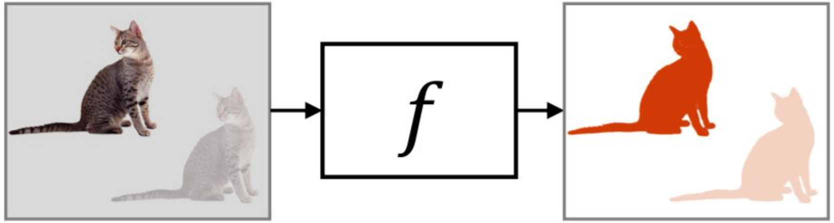
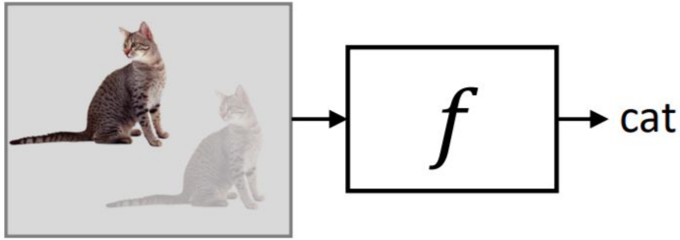
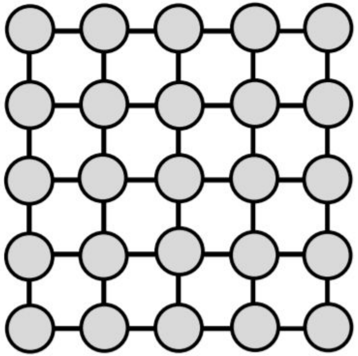


image source: Bronstein et al. “Geometric Deep Learning” Lectures for AMMI, 2021.

Symmetries

Ω





$S_{a,b}$ = shift or translation operator

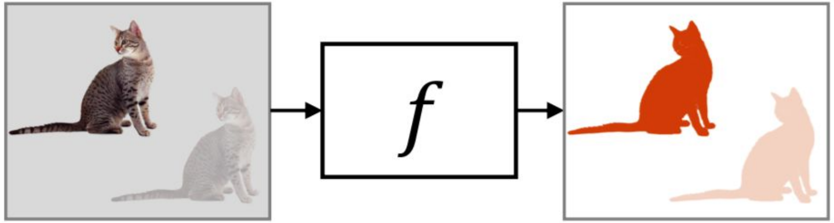
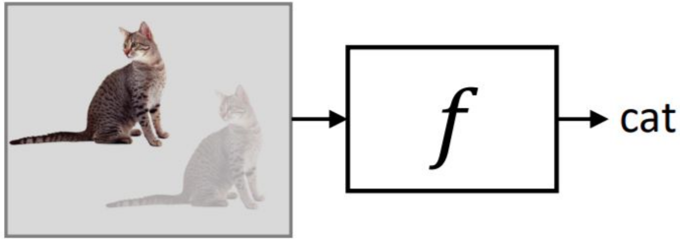
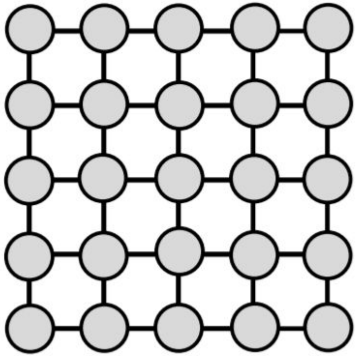


image source: Bronstein et al. “Geometric Deep Learning” Lectures for AMMI, 2021.

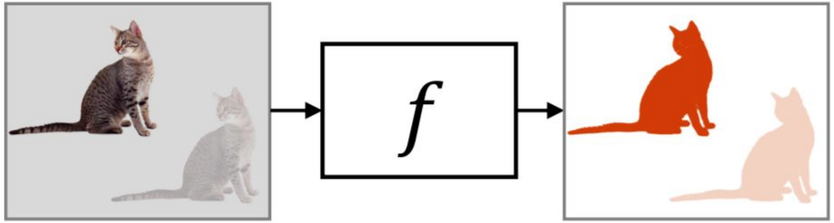
Symmetries

Ω





$$f(x) = f(S_{a,b} \cdot x)$$



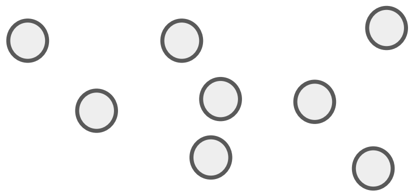
$$f(x) = S_{a,b} \cdot f(x)$$

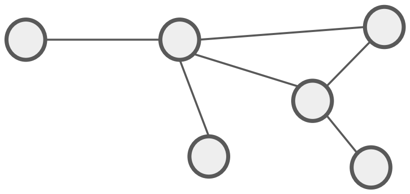
image source: Bronstein et al. "Geometric Deep Learning" Lectures for AMMI, 2021.

Symmetries

Ω

P = permutation of the domain





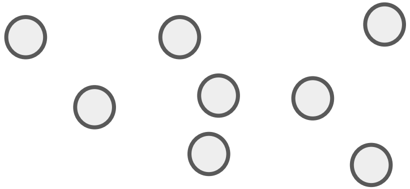
Symmetries

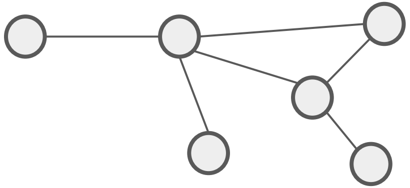
$$\Omega$$

P = permutation of the domain

$$f(x) = f(P \cdot x)$$

$$f(x) = P \cdot f(x)$$





Space of Symmetries $\mathbb{S}$

What structure does the set of all symmetries possess?

Space of Symmetries $\mathbb{S}$

What structure does the set of all symmetries possess?

1. Identity:

$$f(x) = f(i \cdot x) \qquad\qquad i \cdot x = x$$

2. Inverse:

If $f(x) = f(g \cdot x)$ and $g \cdot x = y$ then there should exist $g^{-1} \cdot y = x$

3. Composition

If $f(x) = f(g \cdot x)$ and $f(y) = f(h \cdot y)$ then $f(x) = f((g \cdot h) \cdot x)$

Space of Symmetries $\mathbb{S}$

What structure does the set of all symmetries possess?

1. Identity: $\boxed{i \in \mathbb{S}}$

$$f(x) = f(i \cdot x) \qquad i \cdot x = x$$

2. Inverse: $\boxed{g \in \mathbb{S} \Rightarrow g^{-1} \in \mathbb{S}}$

If $f(x) = f(g \cdot x)$ and $g \cdot x = y$ then there should exist $g^{-1} \cdot y = x$

3. Composition

If $f(x) = f(g \cdot x)$ and $f(y) = f(h \cdot y)$ then $f(x) = f((g \cdot h) \cdot x)$

$$\boxed{g, h \in \mathbb{S} \Rightarrow g \cdot h \in \mathbb{S}}$$

Space of Symmetries $\mathbb{S}$

What is this structure?

What structure does the set of all symmetries possess?

1. Identity: $\boxed{i \in \mathbb{S}}$

   $$f(x) = f(i \cdot x) \qquad i \cdot x = x$$

2. Inverse: $\boxed{g \in \mathbb{S} \Rightarrow g^{-1} \in \mathbb{S}}$

   If $f(x) = f(g \cdot x)$ and $g \cdot x = y$ then there should exist $g^{-1} \cdot y = x$

3. Composition

   If $f(x) = f(g \cdot x)$ and $f(y) = f(h \cdot y)$ then $f(x) = f((g \cdot h) \cdot x)$

   $$\boxed{g, h \in \mathbb{S} \Rightarrow g \cdot h \in \mathbb{S}}$$

Space of Symmetries $\mathbb{S}$

What structure does the set of all symmetries over a domain possess?

Space of Symmetries = Group

$$\mathbb{S} = G$$

Space of Symmetries $\cancel{\mathbb{S}}$ G

Notation

What structure does the set of all symmetries possess?

Space of Symmetries = Group

$$\cancel{\mathbb{S}} = G$$

Notation

Equivalence Relation on $\mathcal{X}(\Omega)$

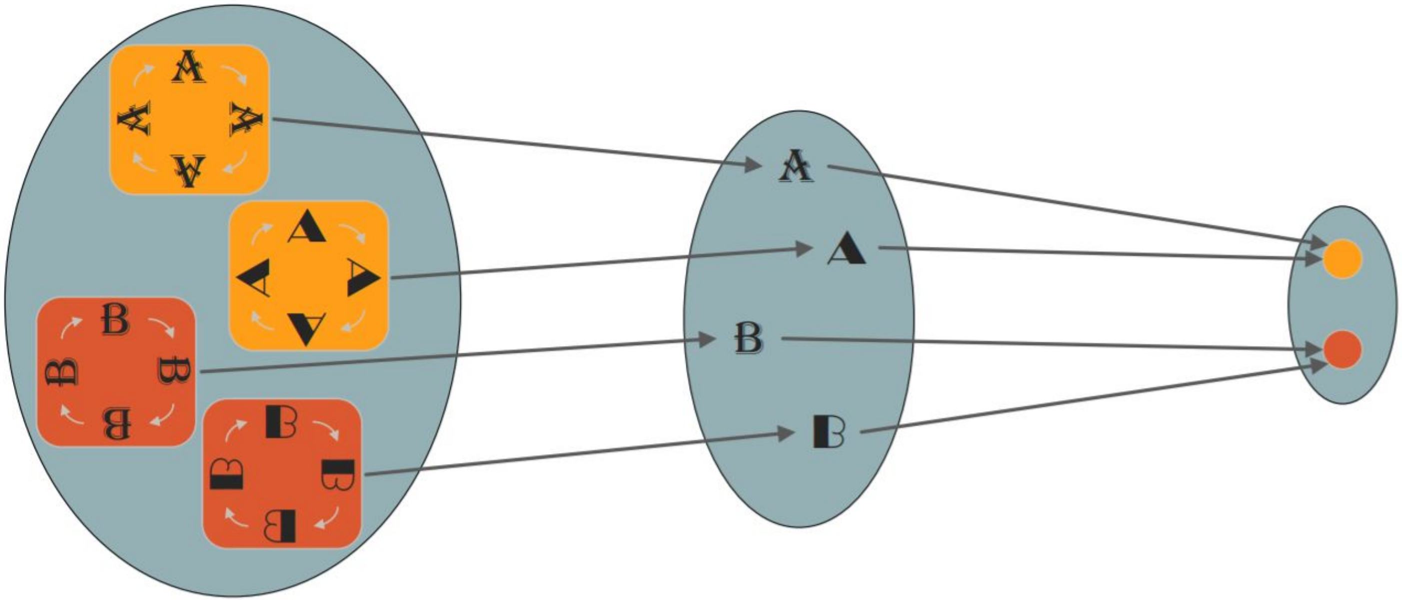
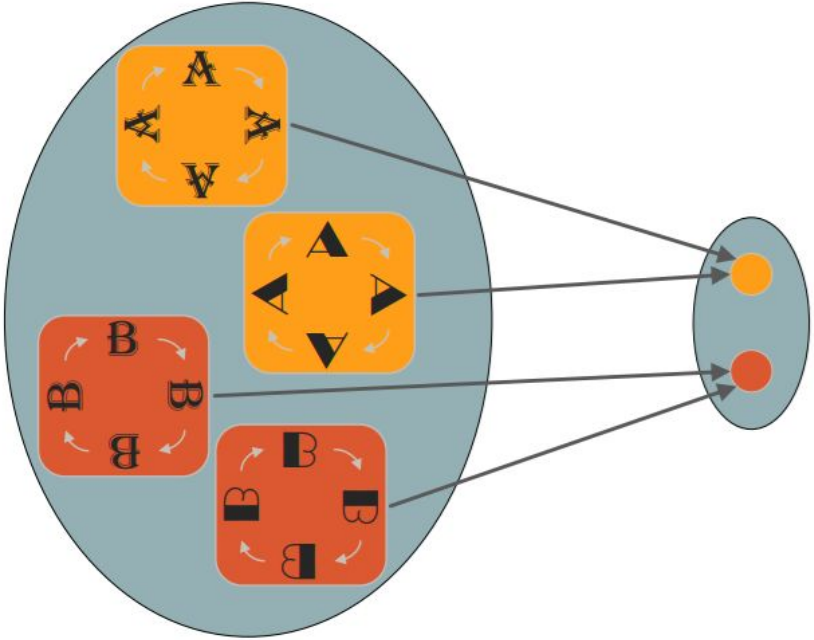


image source: Bronstein et al. “Geometric Deep Learning” Lectures for AMMI, 2021.

Equivalence Relation on $\mathcal{X}(\Omega)$



G-Equivalence

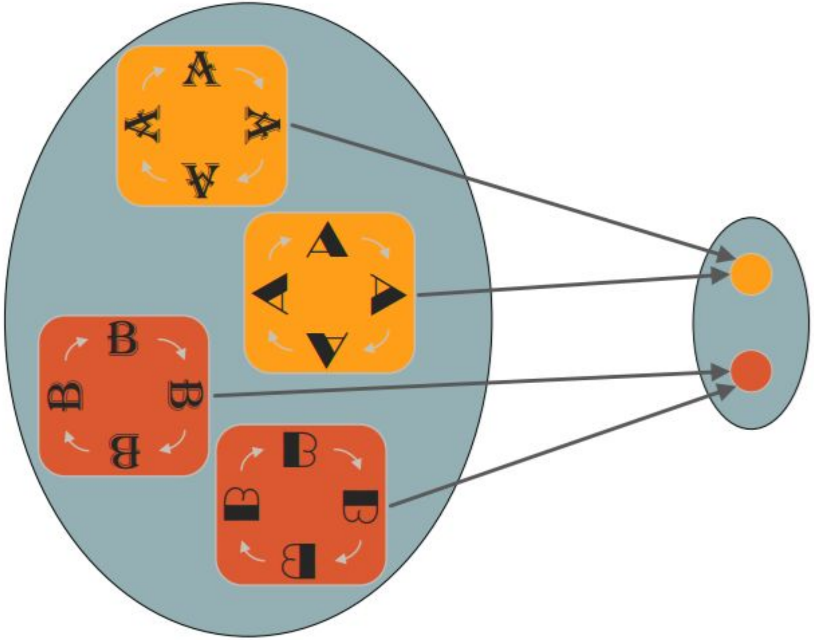
$$x \sim_G y \Leftrightarrow \exists g \in G : gx = y$$

Satisfies the axioms of an equivalence relation:

1. Reflexivity: $x \sim_G x$
   - (Because G contains the identity)
2. Transitivity:
   $$x \sim_G y \wedge y \sim_G z \Leftrightarrow x \sim_G z$$
   - (Because G is closed under composition)
3. Symmetry: $x \sim_G y \Leftrightarrow y \sim_G x$
   - (Because G is closed under inverses)

source: Bronstein et al. “Geometric Deep Learning” Lectures for AMMI, 2021.

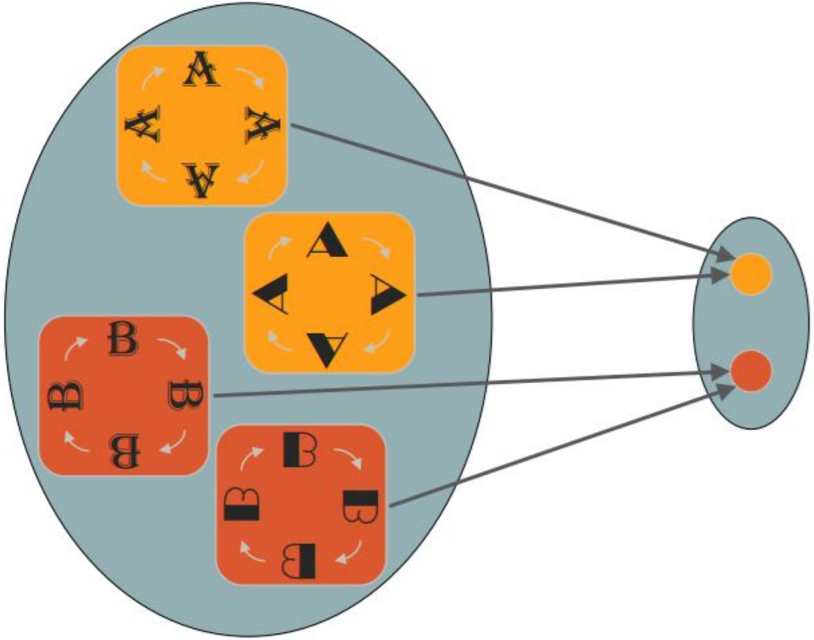
Equivalence Relation on $\mathcal{X}(\Omega)$



If we knew all the symmetries in the input signal space, we wouldn't need any data

image source: Bronstein et al. "Geometric Deep Learning" Lectures for AMMI, 2021.

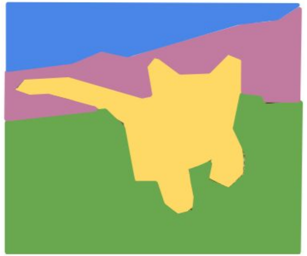
Equivalence Relation on $\mathcal{X}(\Omega)$



More symmetries we exploit, the less data and parameters we will need

image source: Bronstein et al. "Geometric Deep Learning" Lectures for AMMI, 2021.

Adding Structure using Symmetry G

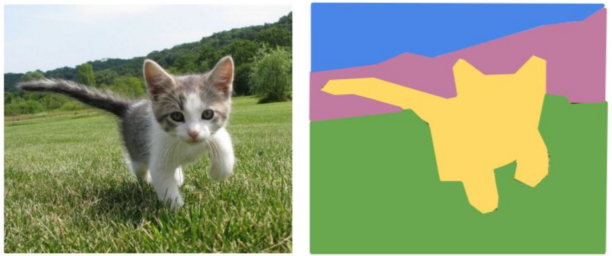
Classification

$$\mathcal{F}_C = \{f : \mathcal{X}(\Omega) \to \mathbb{R}\}$$

Segmentation

$$\mathcal{F}_S = \{f : \mathcal{X}(\Omega) \to \mathcal{X}(\Omega)\}$$

Adding Structure using Symmetry G



Classification

$$\mathcal{F}_C = \{f : \mathcal{X}(\Omega) \to \mathbb{R}\}$$

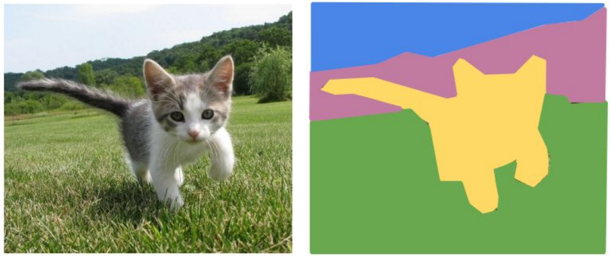
\+ invariance to group action

Segmentation

$$\mathcal{F}_S = \{f : \mathcal{X}(\Omega) \to \mathcal{X}(\Omega)\}$$

\+ equivariance to group action

Adding Structure using Symmetry G



Classification

$$\mathcal{F}_C = \{f : \mathcal{X}(\Omega) \to \mathbb{R} \mid f(g \cdot x) = f(x) \; \forall \, g \in G\}$$

Segmentation

$$\mathcal{F}_S = \{f : \mathcal{X}(\Omega) \to \mathcal{X}(\Omega) \mid f(g \cdot x) = g \cdot f(x) \; \forall \, g \in G\}$$

A Simple Example

Permutation invariant single layer perceptron

Translation Equivariance and Convolution

$$\Omega = [d] \times [d]$$

$$\mathcal{X}(\Omega) = \mathbb{R}^{d \times d}$$

Translation Equivariance and Convolution

$$\Omega = [d] \times [d]$$

$$G = \{S_{k,l} \mid S_{k,l} = \text{shift by } (k, l)\}$$

$$\mathcal{X}(\Omega) = \mathbb{R}^{d \times d}$$

Translation Equivariance and Convolution

$$\Omega = [d] \times [d]$$

$$\mathcal{X}(\Omega) = \mathbb{R}^{d \times d}$$

$$G = \{S_{k,l} \mid S_{k,l} = \text{shift by } (k,l)\}$$

$$(S_{k,l} \cdot x)\,(i,j) = x(i \oplus k, j \oplus l)$$

Translation Equivariance and Convolution

$$\Omega = [d] \times [d]$$

$$G = \{S_{k,l} \mid S_{k,l} = \text{shift by } (k, l)\}$$

$$\mathcal{X}(\Omega) = \mathbb{R}^{d \times d}$$

$$(S_{k,l} \cdot x)(i, j) = x(i \oplus k, j \oplus l)$$

$L : \mathcal{X}(\Omega) \to \mathcal{X}(\Omega)$ is a linear map

Translation Equivariance and Convolution

$$\Omega = [d] \times [d] \qquad G = \{S_{k,l} \mid S_{k,l} = \text{shift by } (k,l)\}$$

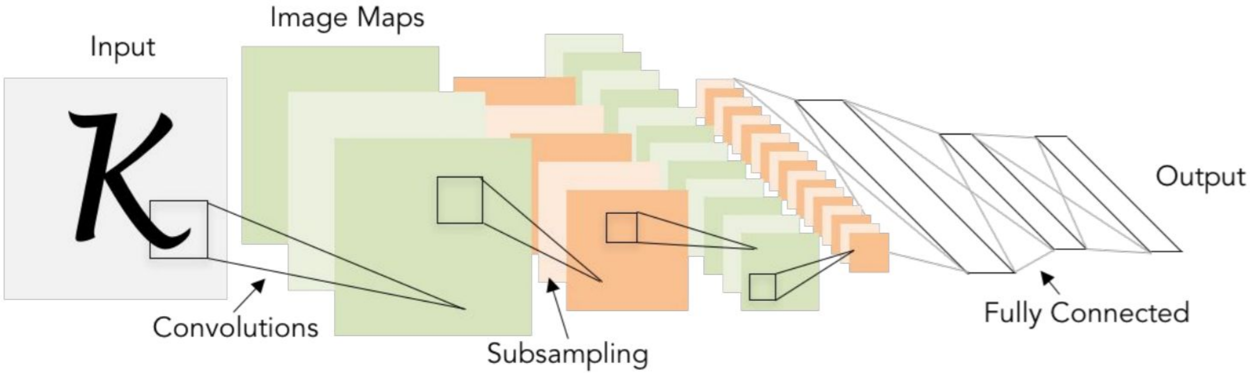
$$\mathcal{X}(\Omega) = \mathbb{R}^{d \times d} \qquad (S_{k,l} \cdot x)\,(i,j) = x(i \oplus k, j \oplus l)$$

$L : \mathcal{X}(\Omega) \to \mathcal{X}(\Omega)$ is a linear map

Theorem

$f(x) = \sigma(L(x))$ is G-equivariant $\Longleftrightarrow$ $L(x)$ is a convolution

What does this mean?



Theorem

$$f^{\text{inv}} : \mathcal{X} \to \mathbb{R}$$

$$f_1^{\text{eq}}, f_2^{\text{eq}}, \ldots f_L^{\text{eq}} : \mathcal{X} \to \mathcal{X}$$

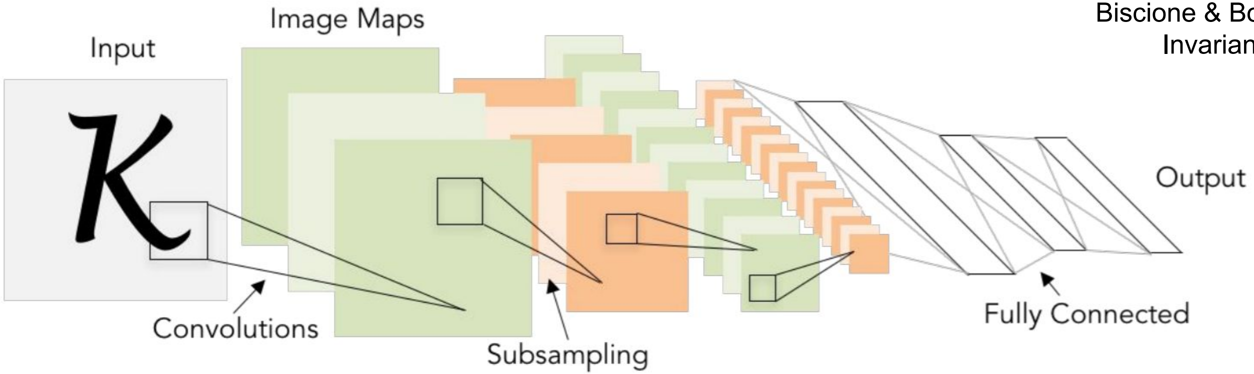
$$f^{\text{inv}} \cdot f_L^{\text{eq}} \cdot f_{L-1}^{\text{eq}} \cdots f_1^{\text{eq}} : \mathcal{X} \to \mathbb{R}$$

is G-invariant

What does this mean?

CNNs are not strictly translation invariant!

Biscione & Bowers "Learning Translation Invariance in CNNs" NeurIPS 2020



Theorem

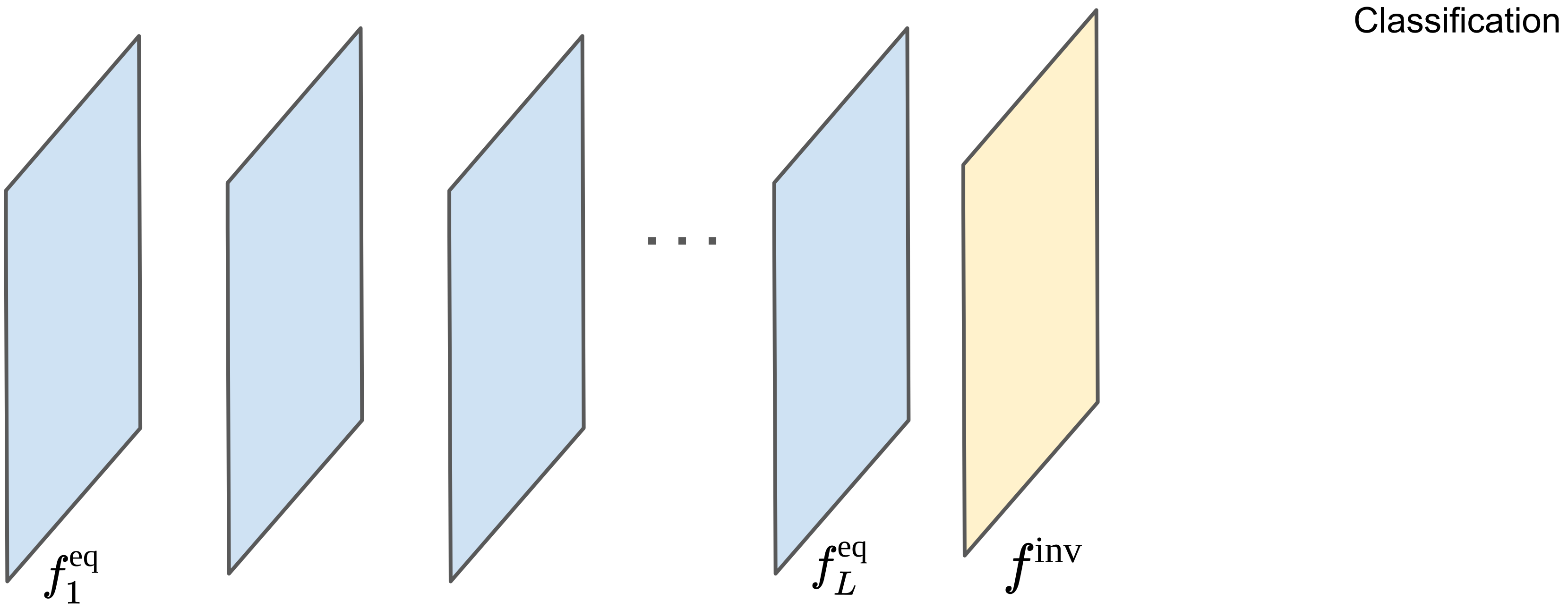
$$f^{\mathrm{inv}} : \mathcal{X} \to \mathbb{R}$$

$$f_1^{\mathrm{eq}}, f_2^{\mathrm{eq}}, \ldots f_L^{\mathrm{eq}} : \mathcal{X} \to \mathcal{X}$$

$$f^{\mathrm{inv}} \cdot f_L^{\mathrm{eq}} \cdot f_{L-1}^{\mathrm{eq}} \cdots f_1^{\mathrm{eq}} : \mathcal{X} \to \mathbb{R}$$

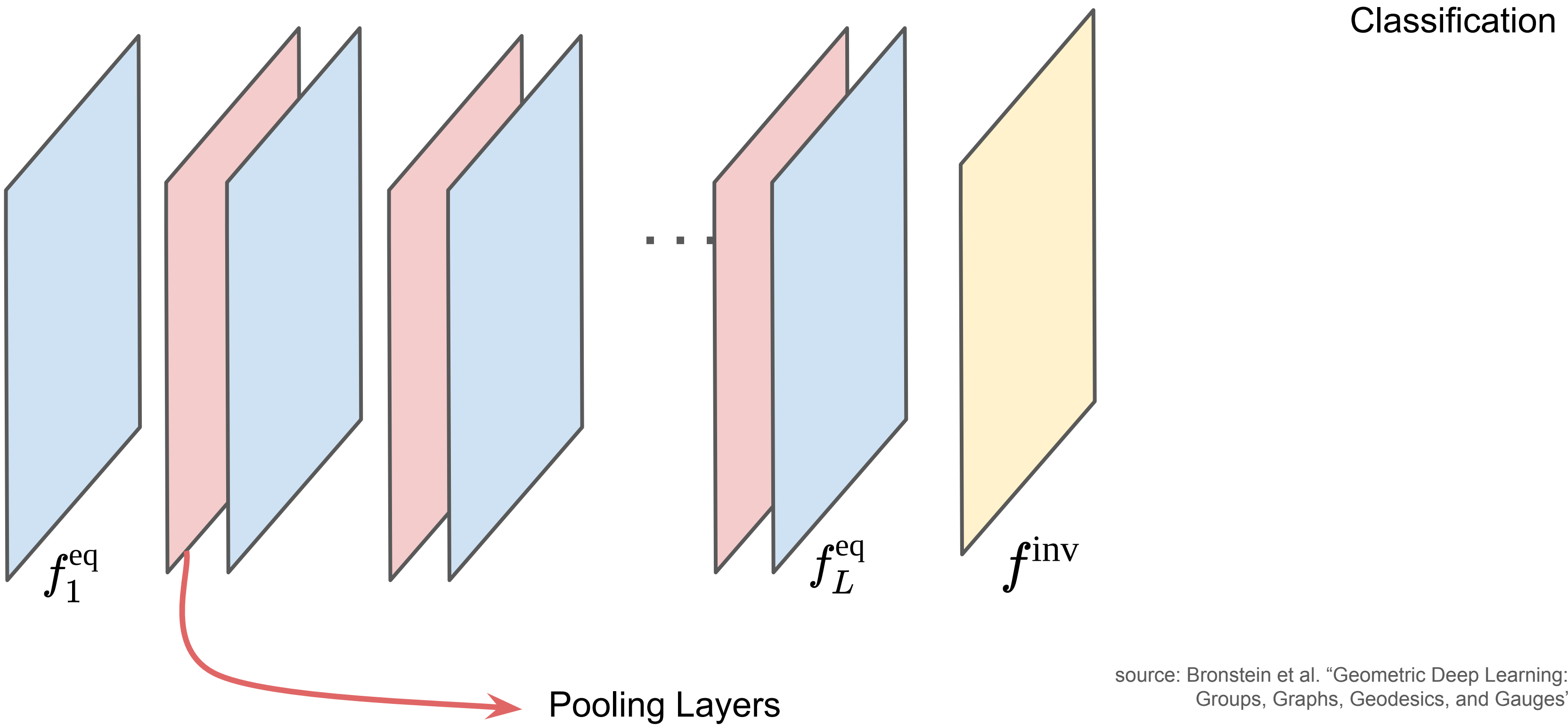
is G-invariant

Geometric Deep Learning Blueprint



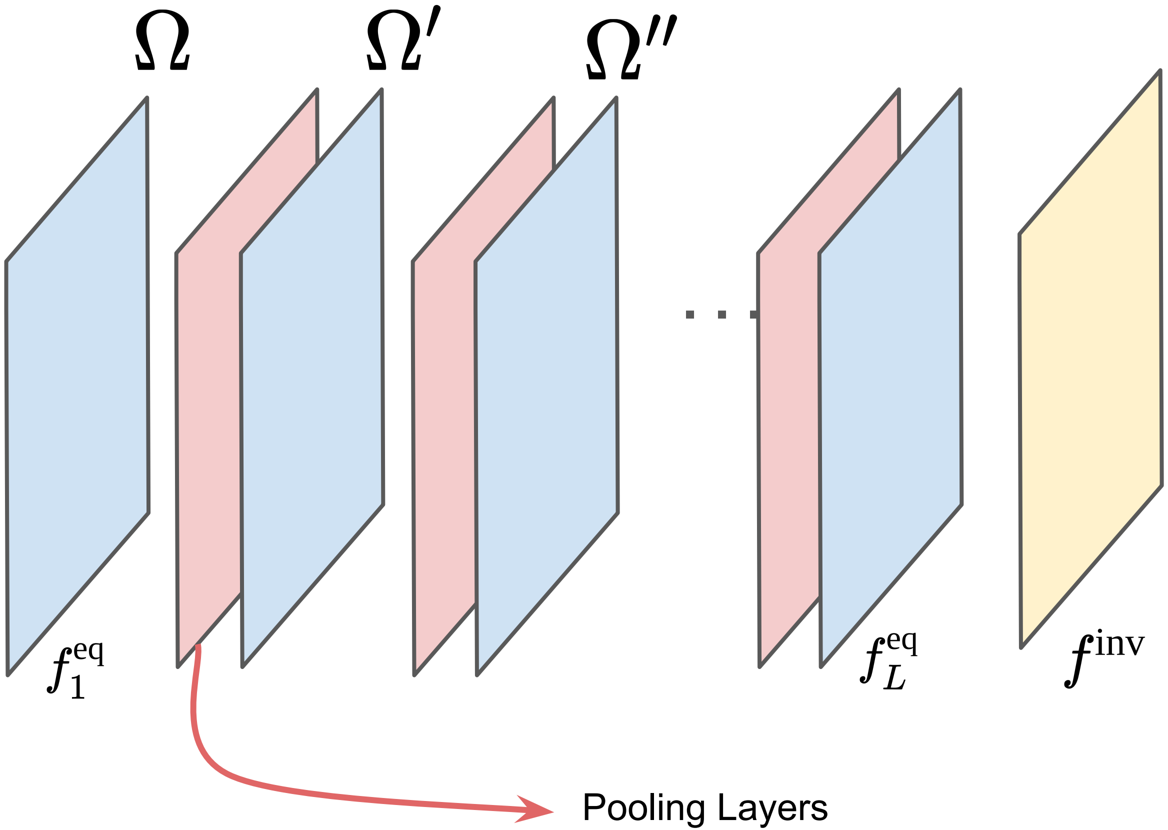
source: Bronstein et al. “Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges” 2021.

Geometric Deep Learning Blueprint



source: Bronstein et al. “Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges” 2021.

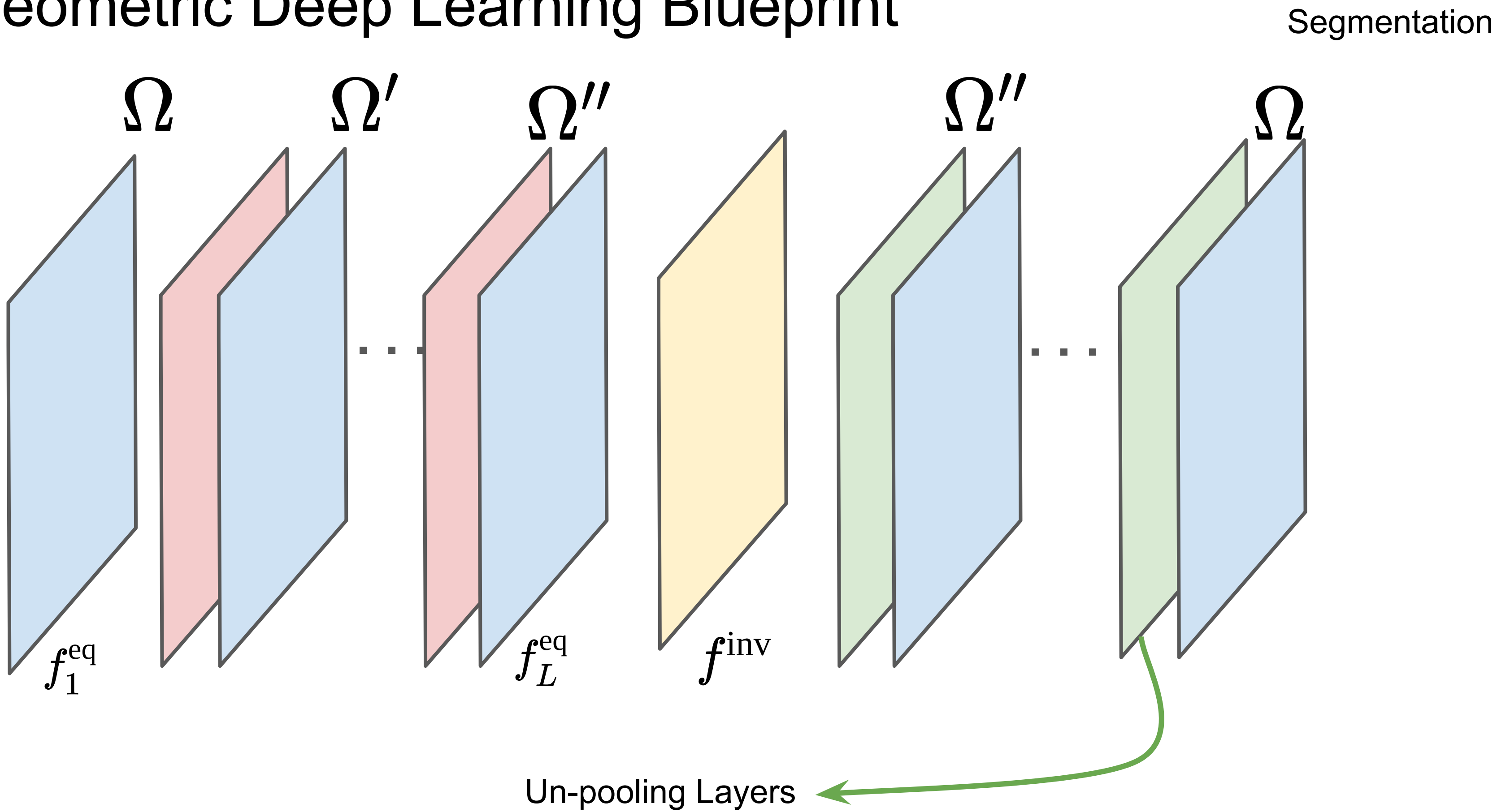
Geometric Deep Learning Blueprint



Classification

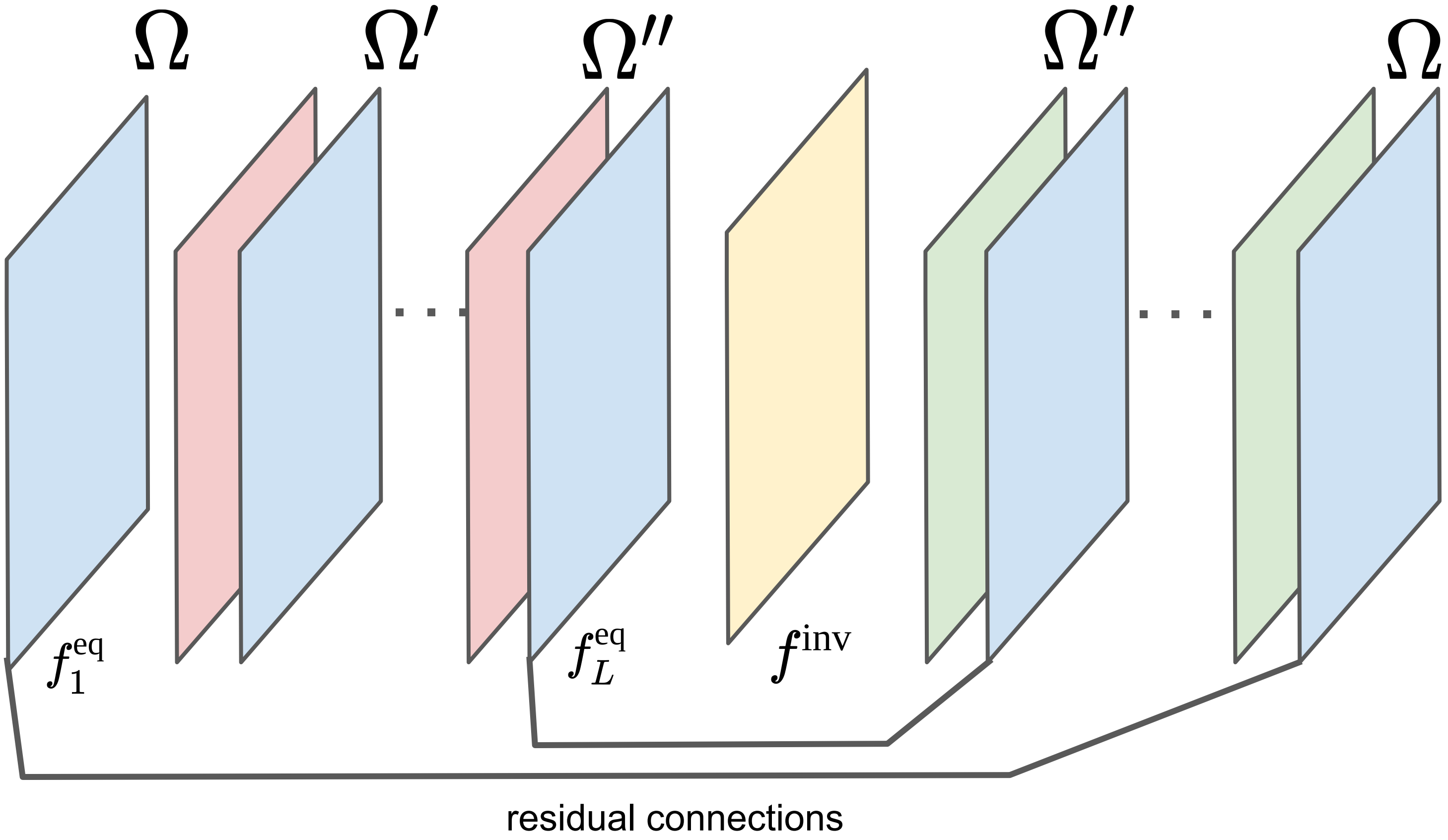
source: Bronstein et al. “Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges” 2021.

Geometric Deep Learning Blueprint



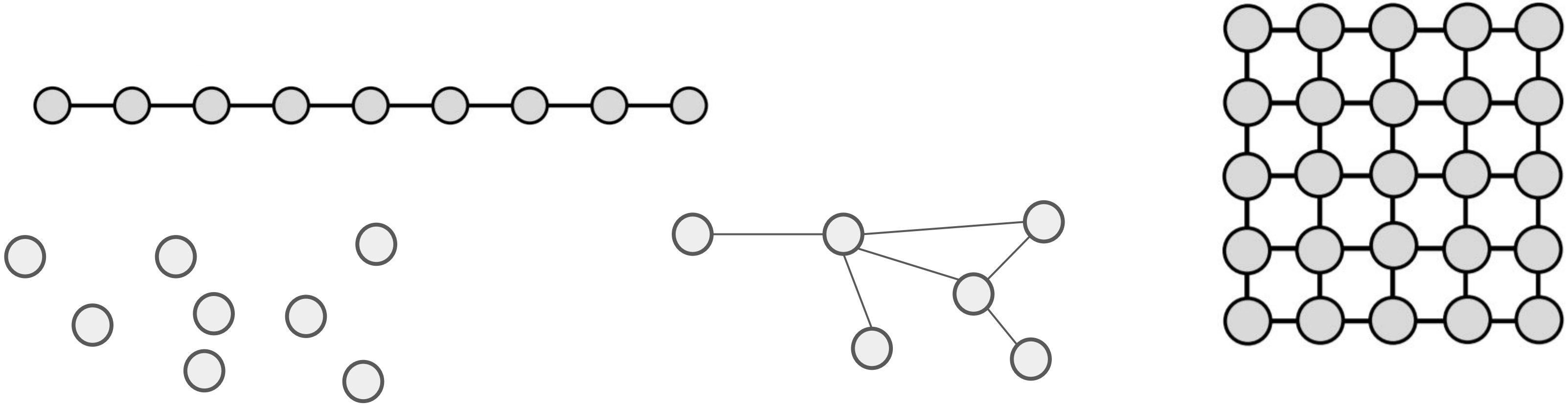
Geometric Deep Learning Blueprint

Segmentation



Using the Blueprint

Suffices to find invariant and equivariant functions on different domains



Sets

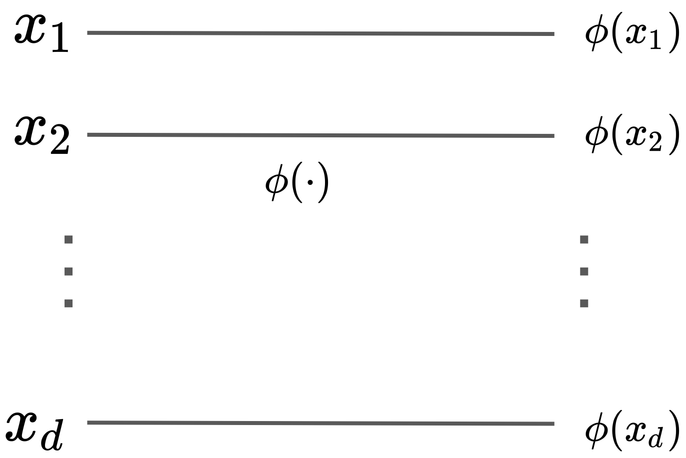
Equivariance over Sets

$\Omega = [d]$ $\qquad$ $\mathcal{X}(\Omega) = \mathbb{R}^d$ $\qquad$ $G = \{P \mid P = d \times d \text{ permutation matrix}\}$

$$f(P \cdot x) = P \cdot f(x)$$

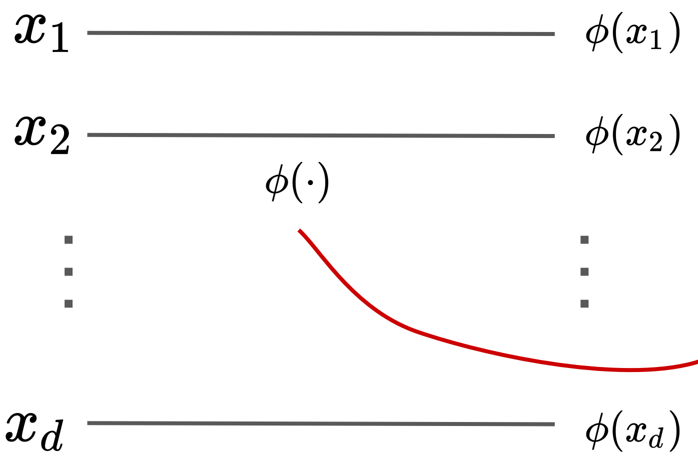
Equivariance over Sets

$\Omega = [d]$ $\qquad \mathcal{X}(\Omega) = \mathbb{R}^d$ $\qquad G = \{P \mid P = d \times d \text{ permutation matrix}\}$



Equivariance over Sets

$\Omega = [d]$ $\qquad$ $\mathcal{X}(\Omega) = \mathbb{R}^d$ $\qquad$ $G = \{P \mid P = d \times d \text{ permutation matrix}\}$



Can be a permutation invariant function of all the inputs

Recall: Point Transformer

Basic version

$$x'_j = \sum_{i \in N(x_j)} \rho(\phi(x_j)^T \psi(x_i)) \cdot \alpha(x_i)$$

Incorporating point feature + location; and using vector for attention

$$x'_j = \sum_{i \in N(x_j)} \rho[\beta(\phi(x_j), \psi(x_i)) + \delta(p_j - p_i)] \odot \alpha(x_i)$$

function other than
dot product

position of points

Zhao et al. "Point Transformer" 2020

Invariance over Sets

$\Omega = [d]$ $\quad$ $\mathcal{X}(\Omega) = \mathbb{R}^d$ $\quad$ $G = \{P \mid P = d \times d \text{ permutation matrix}\}$

$$f(P \cdot x) = f(x)$$

Invariance over Sets

$\Omega = [d]$ $\qquad \mathcal{X}(\Omega) = \mathbb{X}^d$ $\qquad G = \{P \mid P = d \times d \text{ permutation matrix}\}$

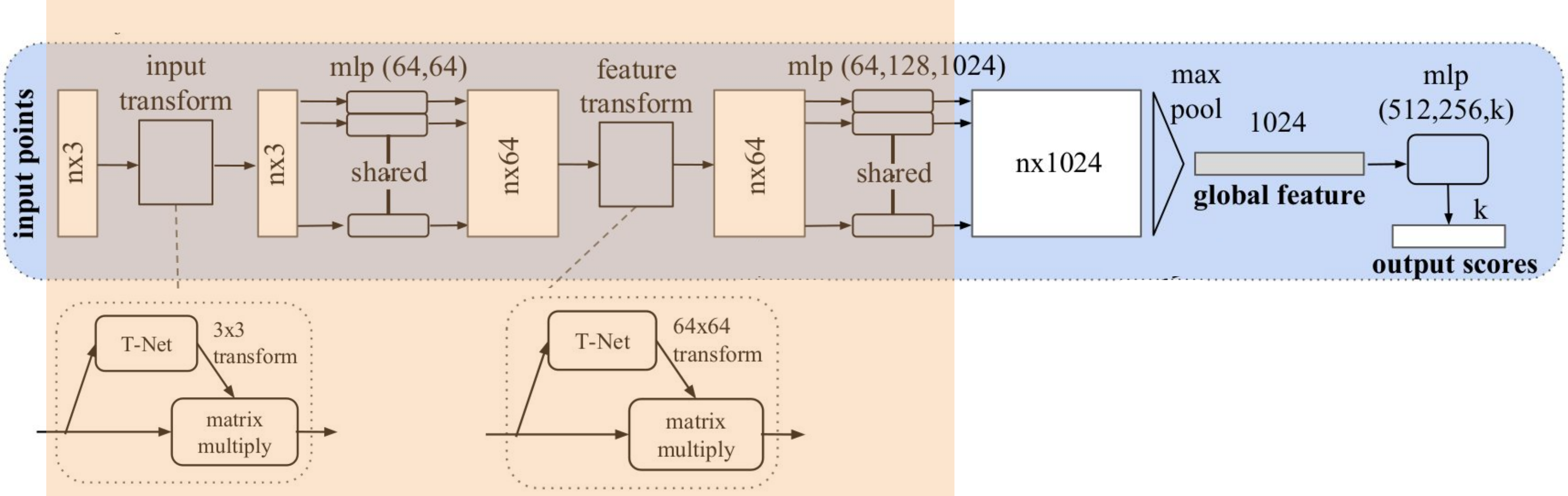
countable set

Theorem:

$f : \mathcal{X}(\Omega) \to \mathbb{R}$ is G-invariant $\quad \Longleftrightarrow \quad f(x) = \psi\left(\sum_{i=1}^{d} \phi(x_i)\right)$ $\exists\, \phi, \psi$

Zaheer et al. "Deep Sets" NeurIPS 2017

Recall: PointNet Architecture



$$f(\{x_1, x_2, \ldots x_n\}) = \max\{h(x_i), h(x_2), \ldots h(x_n)\}$$

Qi et al. "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation" CVPR 2017

Simple Example

Permutation equivariant single layer perceptron

Question

Why not use the blueprint with permutation invariant and equivariant single layer perceptron?

Backup

Error Decomposition