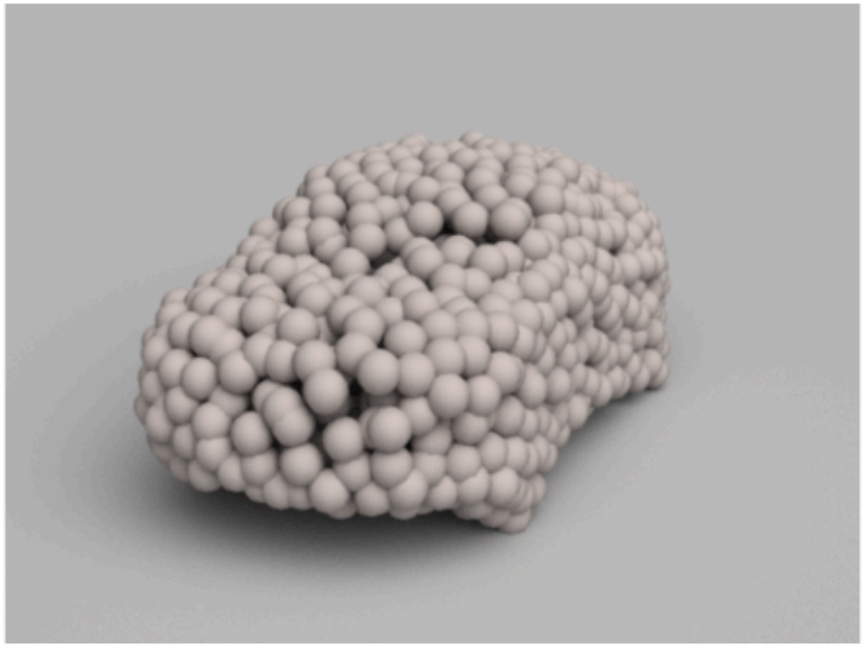
3D Deep Learning on Geometric Forms



Hao Su

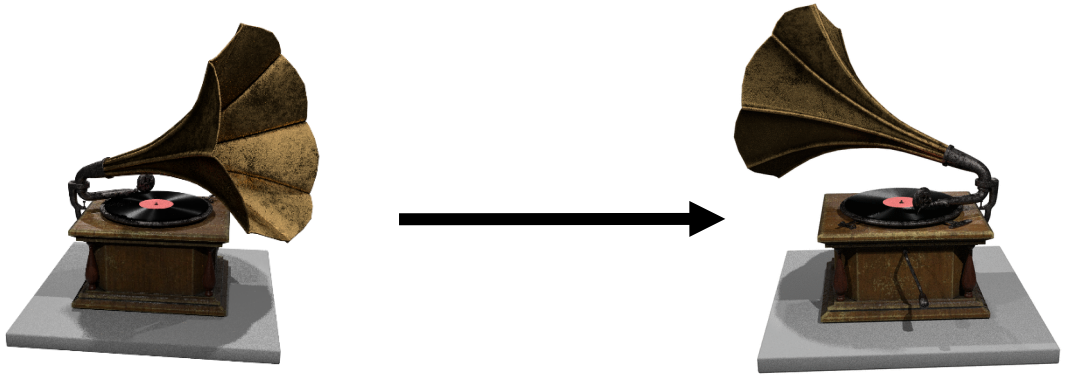
Stanford University

Many 3D representations are available

Candidates:

- multi-view images
- depth map
- volumetric
- polygonal mesh
- point cloud
- primitive-based CAD models

3D representation



Novel view image synthesis

[Su et al., ICCV15]
[Dosovitskiy et al., ECCV16]

Candidates:

multi-view images

depth map

volumetric

polygonal mesh

point cloud

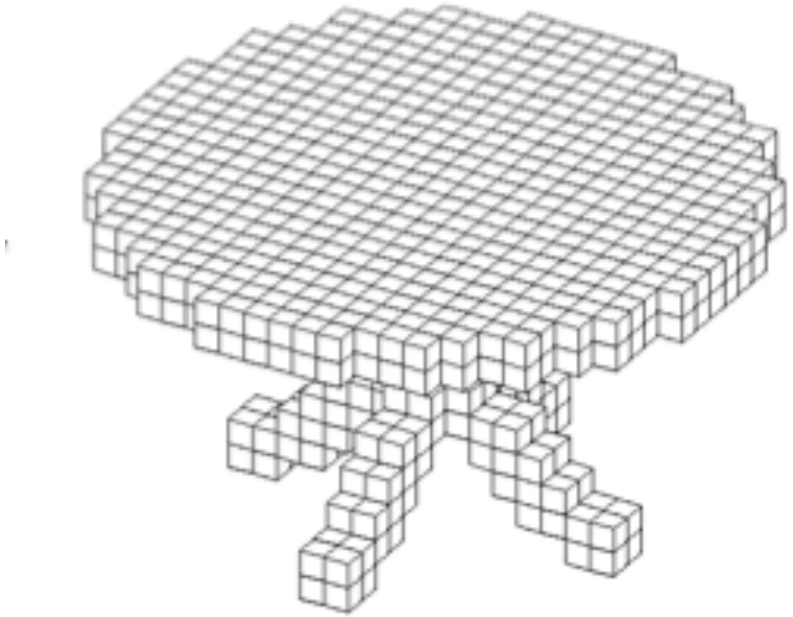
primitive-based CAD models

3D representation

Candidates:

- multi-view images
- **depth map**
- volumetric
- polygonal mesh
- point cloud
- primitive-based CAD models

3D representation



Candidates:

multi-view images

depth map

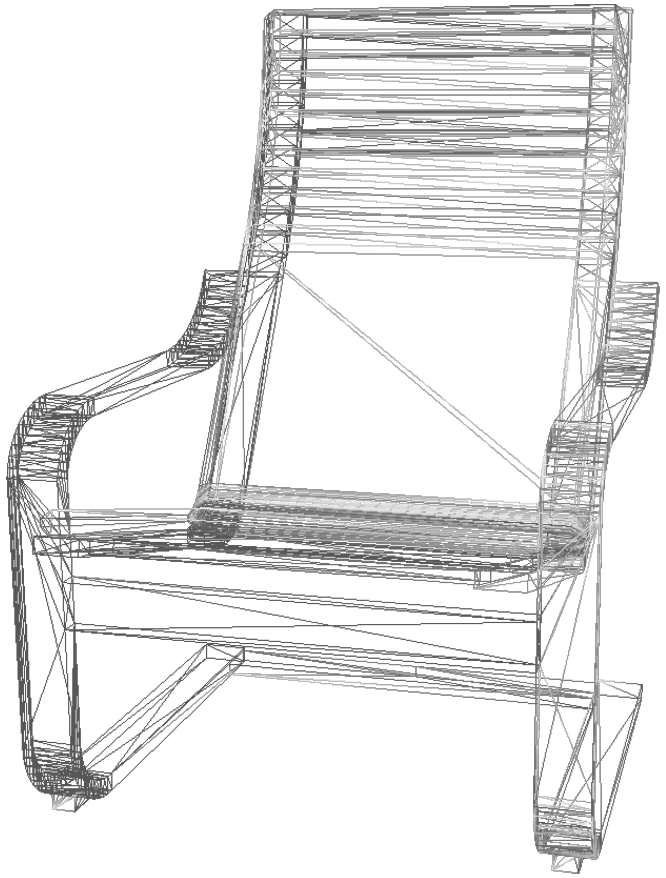
volumetric

polygonal mesh

point cloud

primitive-based CAD models

3D representation



Candidates:

multi-view images

depth map

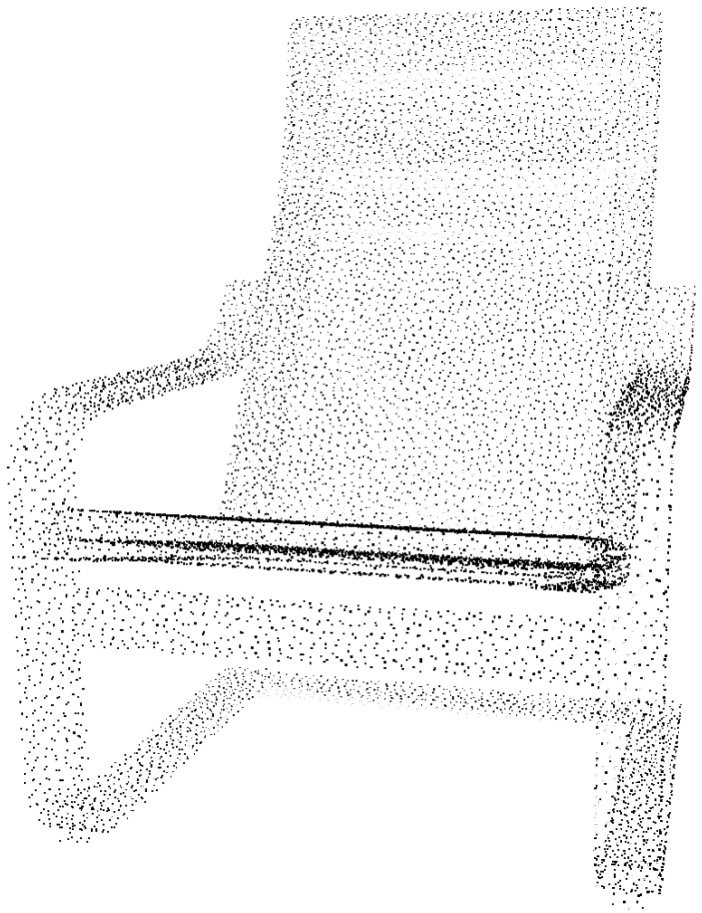
volumetric

polygonal mesh

point cloud

primitive-based CAD models

3D representation



Candidates:

multi-view images

depth map

volumetric

polygonal mesh

point cloud

primitive-based CAD models

3D representation

a chair assembled by cuboids

Candidates:

multi-view images

depth map

volumetric

polygonal mesh

point cloud

primitive-based CAD models

Two groups of representations

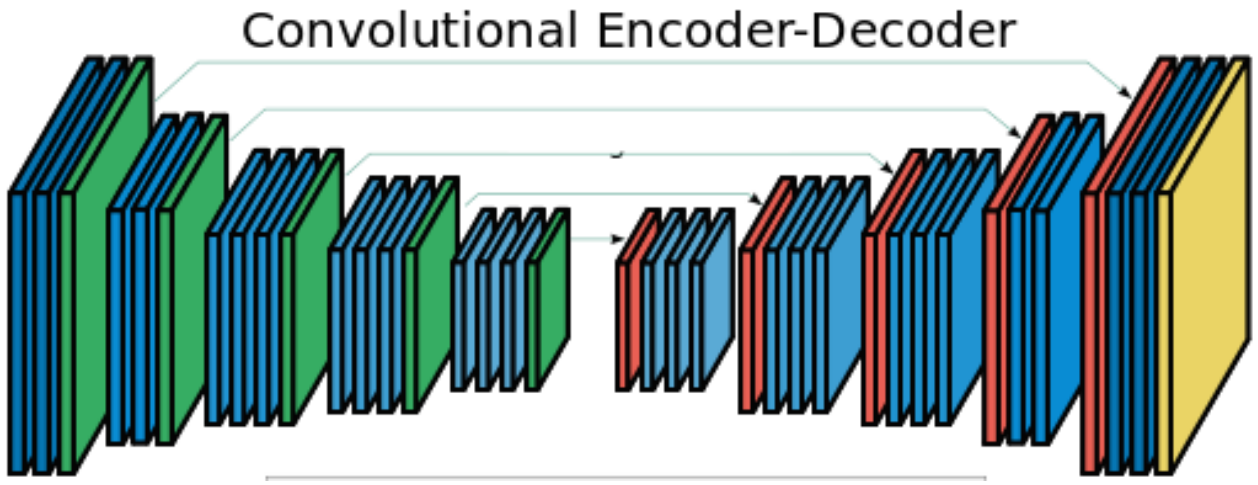
Rasterized form (regular grids)

Geometric form (irregular)

Candidates:

- multi-view images
- depth map
- volumetric
- polygonal mesh
- point cloud
- primitive-based CAD models

Extant 3D DNNs work on grid-like representations



Candidates:

- multi-view images
- depth map
- volumetric
- polygonal mesh
- point cloud
- primitive-based CAD models

Ideally, a 3D representation should be

Friendly to learning

- easily formulated as the input/output of a neural network
- fast forward-/backward- propagation
- etc.

Ideally, a 3D representation should be

Friendly to learning

- easily formulated as the input/output of a neural network
- fast forward-/backward- propagation
- etc.

Flexible

- can precisely model a great variety of shapes
- etc.

Ideally, a 3D representation should be

Friendly to learning

- easily formulated as the output of a neural network
- fast forward-/backward- propagation
- etc.

Flexible

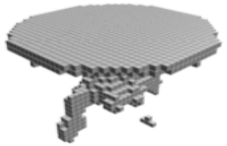
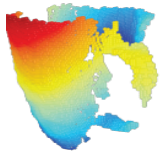
- can precisely model a great variety of shapes
- etc.

Geometrically manipulable for networks

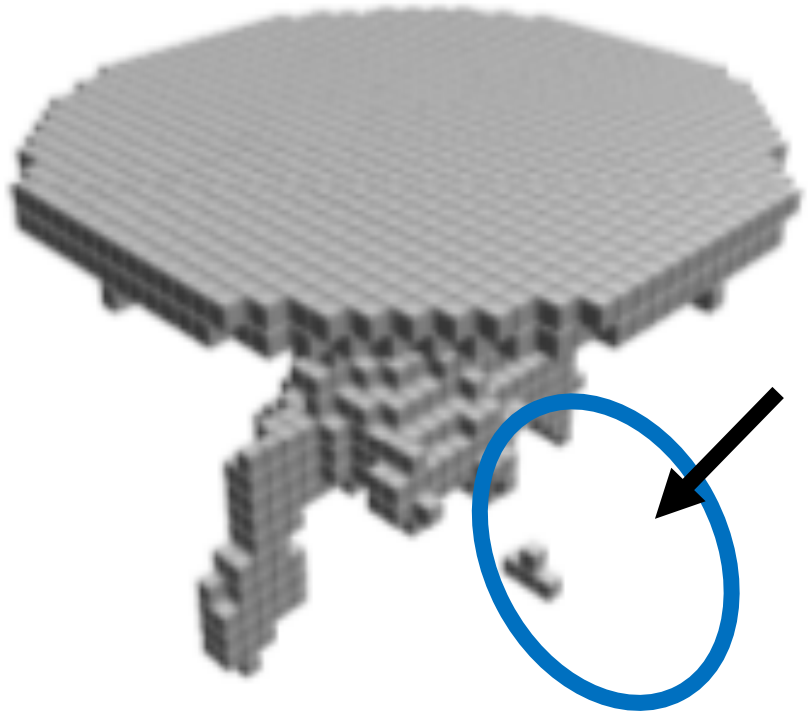
- geometrically deformable, interpolable and extrapolable for networks
- convenient to impose structural constraints
- etc.

Others

The problem of grid representations

	Affability to learning	Flexibility	Geometric manipulability
Multi-view images 	✓	✓	✗
Volumetric occupancy 	✗ Expensive to compute: $O(N^3)$	✓	✗
Depth map 	✓	✗ Cannot model “back side”	✓

Typical artifacts of volumetric reconstruction



Missing or extra thin structures

Volumes are hard for **the network** to rotate / deform / interpolate

Learn to analyze / generate Geometric Forms?

Candidates:

Rasterized form (regular grids)

- multi-view images
- depth map
- volumetric

Geometric form (irregular)

- polygonal mesh
- ★ **point cloud**
- ★ **primitive-based CAD models**

Outline

Motivation

3D point cloud / CAD model reconstruction

3D point cloud analysis, e.g., segmentation

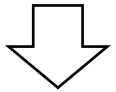
3D point clouds

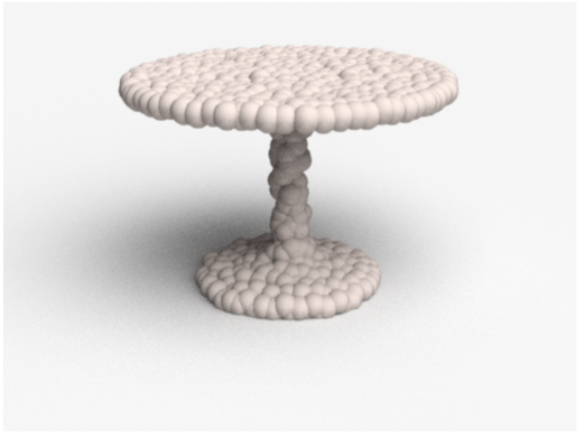
A dual formulation of occupancy

✓ Flexibility

✓ Geometric manipulability

? Affability to learning





Eulerian

Prob. distribution

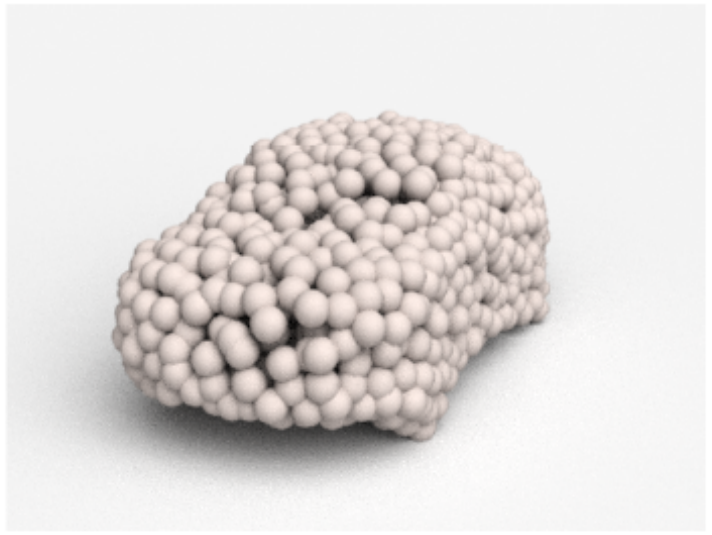
Volumetric occupancy

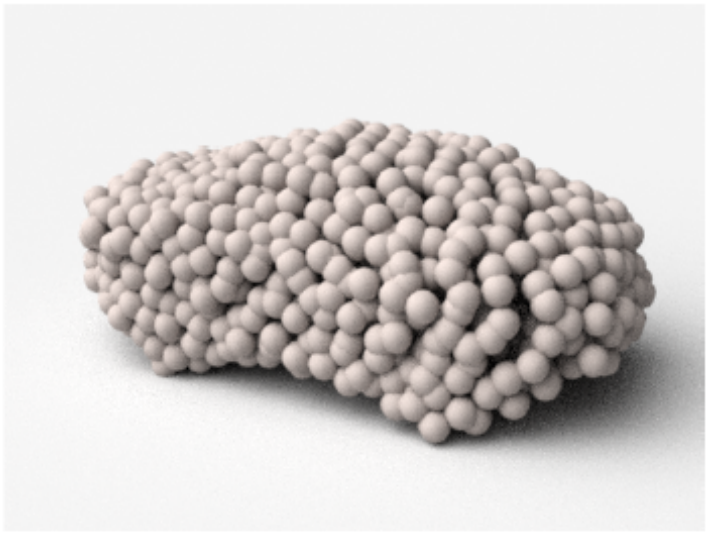
Lagrangian

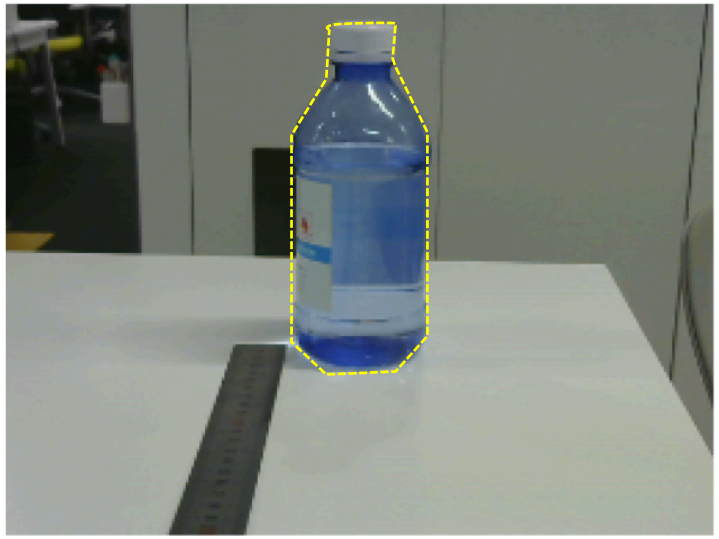
Particle filters

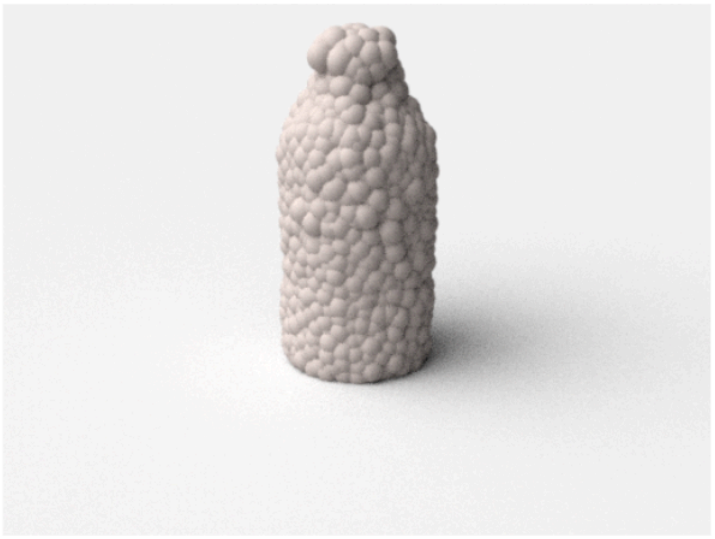
Point clouds

Result: 3D reconstruction from real Images





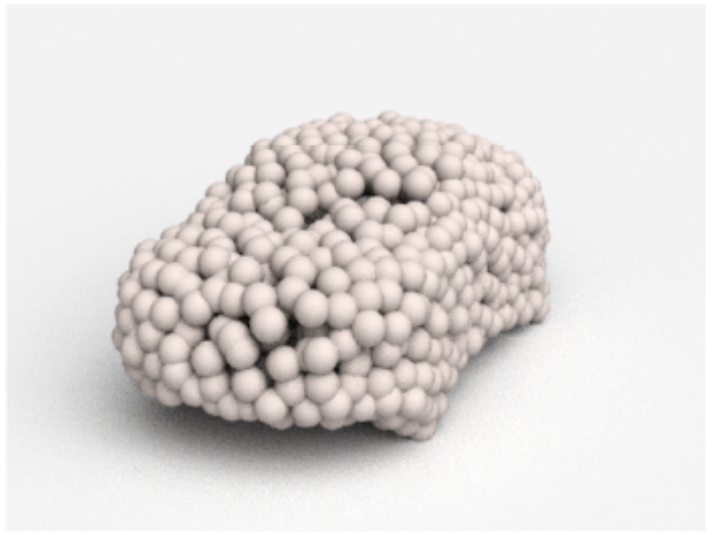


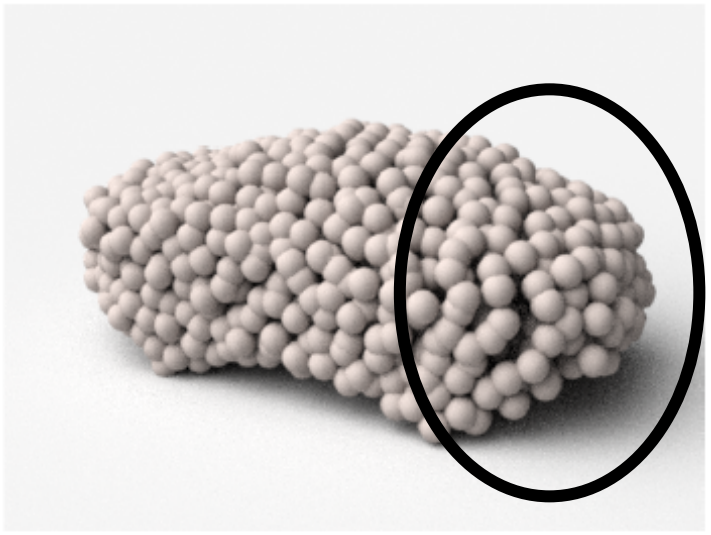


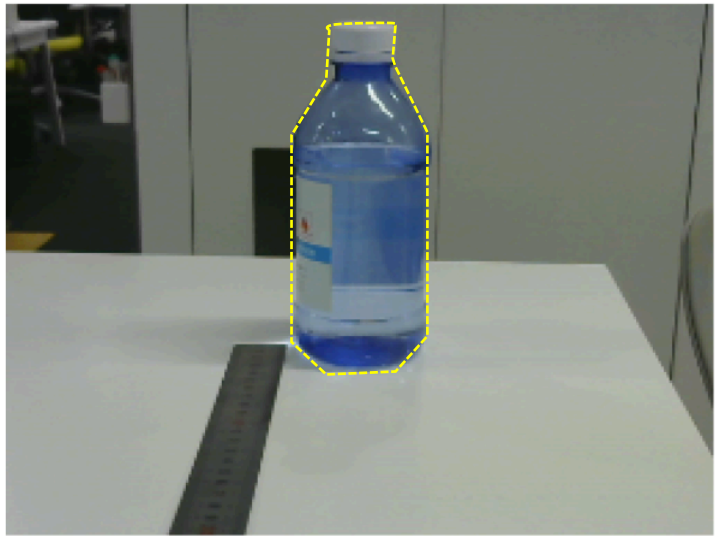
Input

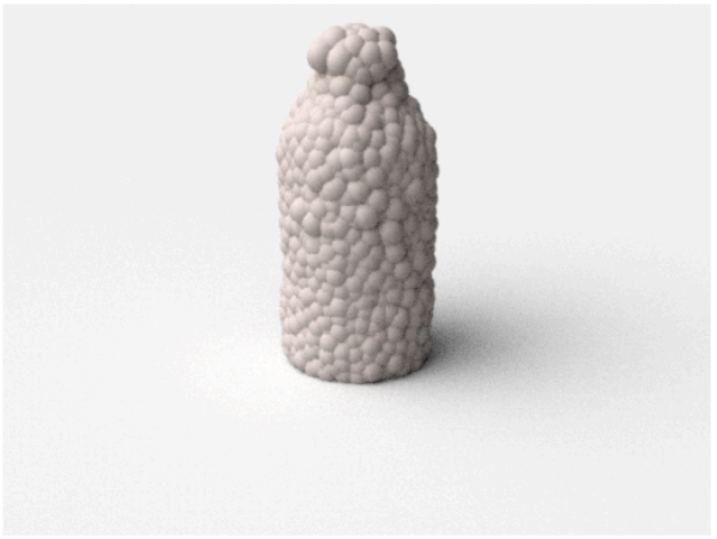
Reconstructed 3D point cloud

Result: 3D reconstruction from real Images









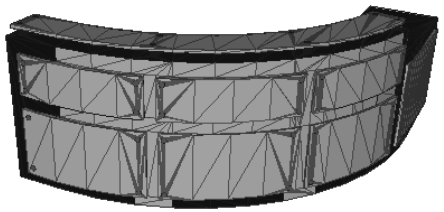
Input

Reconstructed 3D point cloud

An end-to-end synthesis-for-learning system

Image

rendering



3D model

sampling

$$\left\{ \begin{array}{c} (x'_1, y'_1, z'_1) \\ (x'_2, y'_2, z'_2) \\ \cdots \\ (x'_n, y'_n, z'_n) \end{array} \right\}$$

Groundtruth point cloud

An end-to-end learning system

Image

Deep Neural Network

Predicted set

$$\left\{ \begin{array}{c} (x_1, y_1, z_1) \\ (x_2, y_2, z_2) \\ \dots \\ (x_n, y_n, z_n) \end{array} \right\}$$

$$\left\{ \begin{array}{c} (x'_1, y'_1, z'_1) \\ (x'_2, y'_2, z'_2) \\ \dots \\ (x'_n, y'_n, z'_n) \end{array} \right\}$$

Groundtruth point cloud

An end-to-end learning system

Image

Deep Neural Network

Predicted set

$$\left\{ \begin{array}{c} (x_1, y_1, z_1) \\ (x_2, y_2, z_2) \\ \dots \\ (x_n, y_n, z_n) \end{array} \right\}$$

Point Set Distance

$$\left\{ \begin{array}{c} (x'_1, y'_1, z'_1) \\ (x'_2, y'_2, z'_2) \\ \dots \\ (x'_n, y'_n, z'_n) \end{array} \right\}$$

Groundtruth point cloud

An end-to-end learning system

Image

Deep Neural Network

Predicted set

$$\left\{ \begin{array}{c} (x_1, y_1, z_1) \\ (x_2, y_2, z_2) \\ \dots \\ (x_n, y_n, z_n) \end{array} \right\}$$

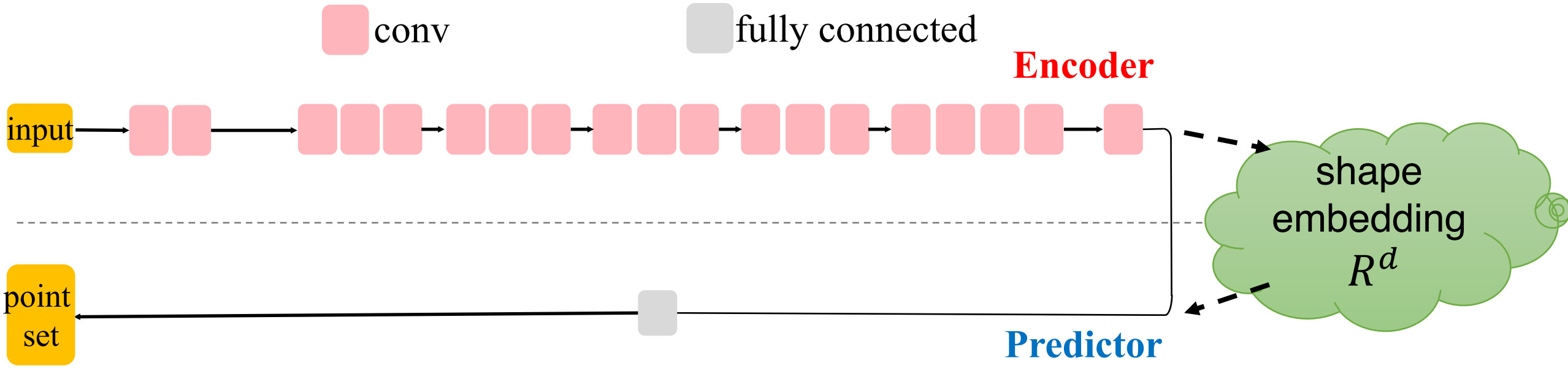
Point Set Distance

$$\left\{ \begin{array}{c} (x'_1, y'_1, z'_1) \\ (x'_2, y'_2, z'_2) \\ \dots \\ (x'_n, y'_n, z'_n) \end{array} \right\}$$

Groundtruth point cloud

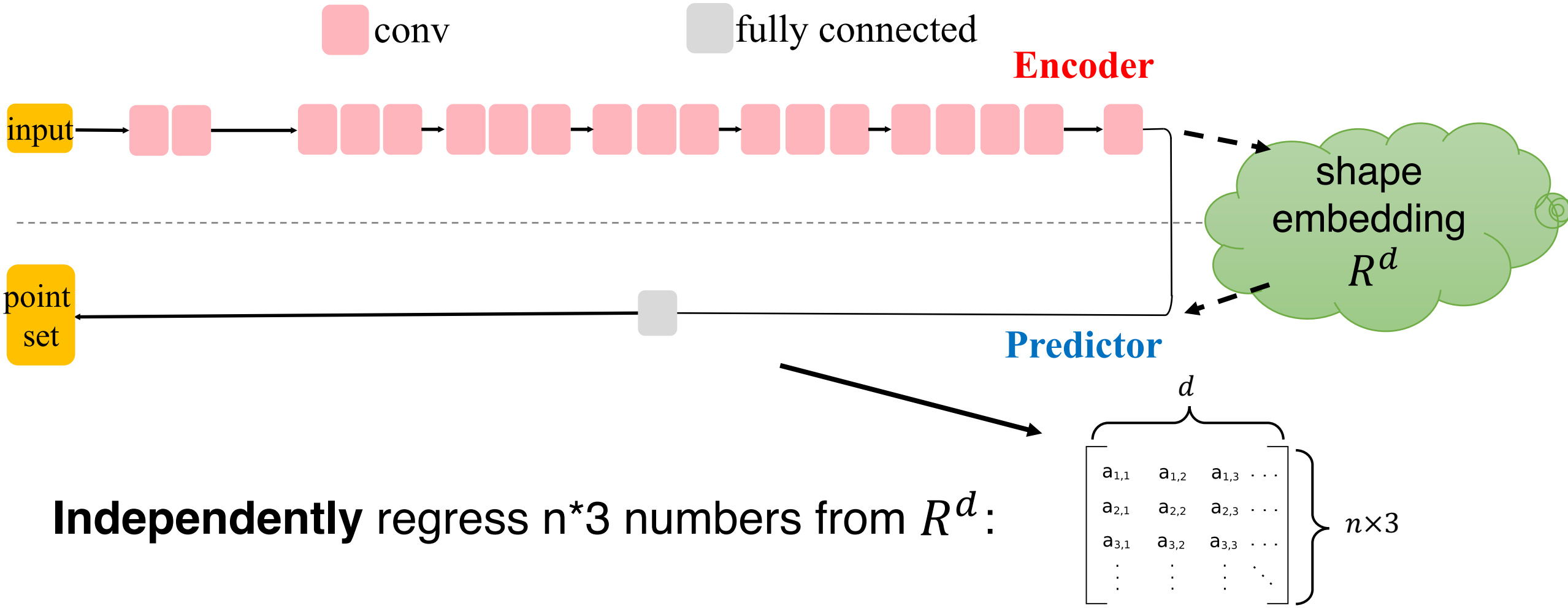
Network architecture: Vanilla version

Fully connected layer as predictor in standard classification network



Network architecture: Vanilla version

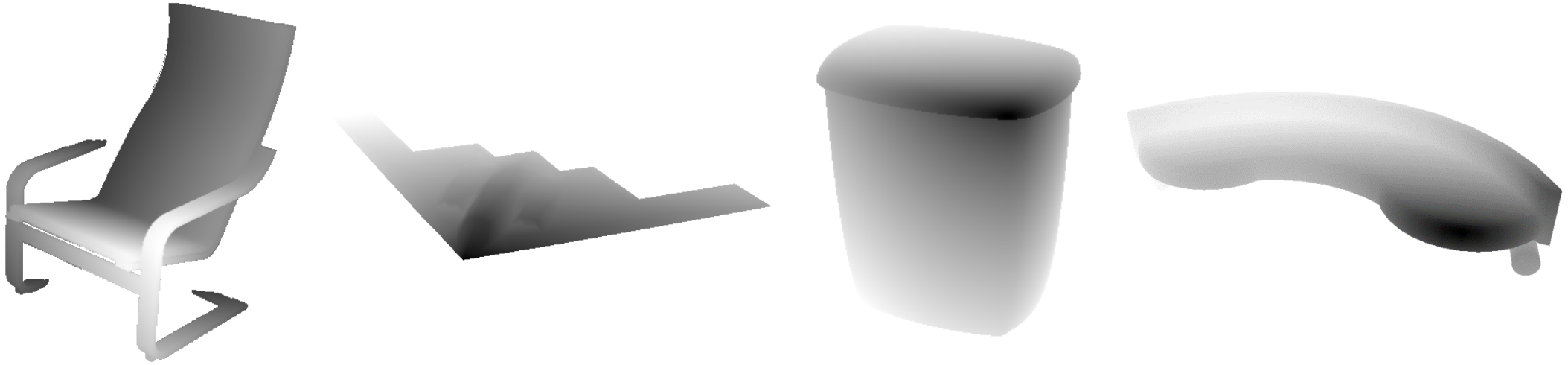
Fully connected layer as predictor in standard classification network



Independently regress n*3 numbers from R^d:

Natural statistics of geometry

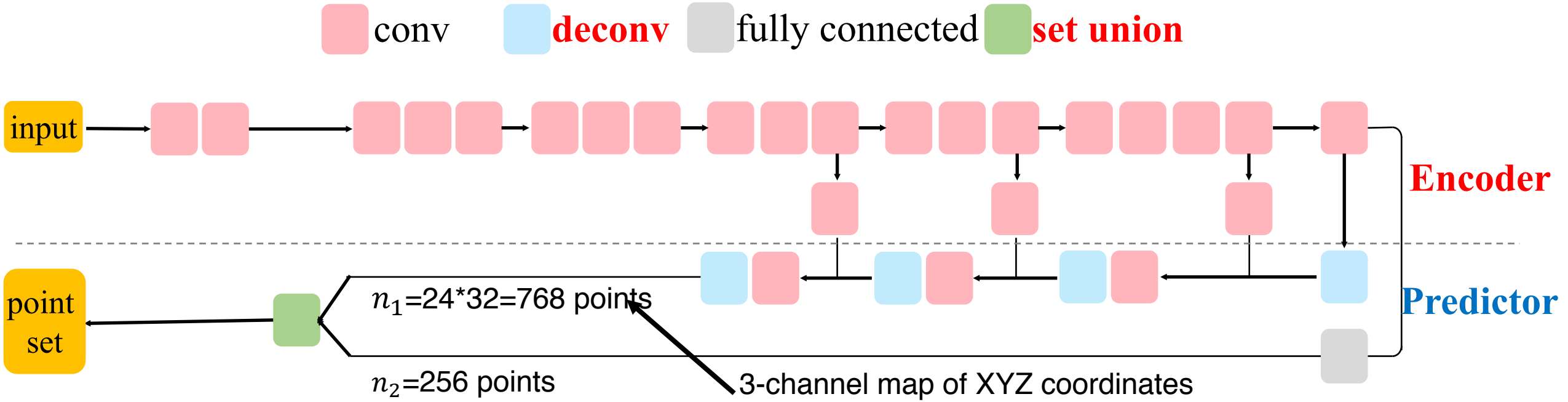
- Many objects, especially man-made objects, contain large **smooth** surfaces



- Deconvolution can generate locally smooth textures for images

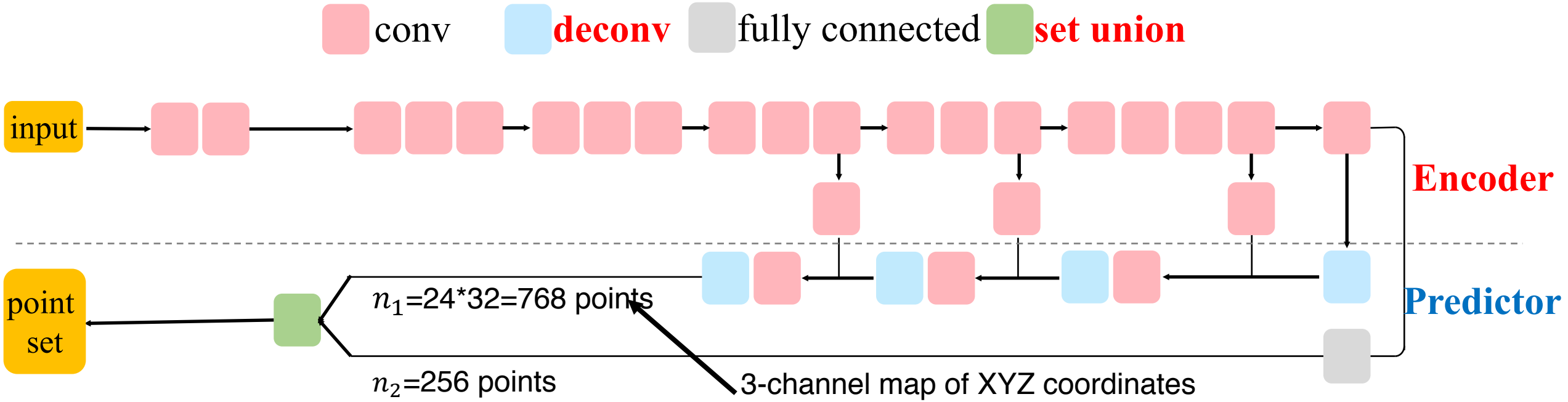
Network architecture: Output from deconv branch

Two branch version



Network architecture: Output from deconv branch

Two branch version



$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_1 \\ \mathbf{C}_2 \end{bmatrix} \qquad \mathbf{C}_1 \in \mathbb{R}^{n_1 \times 3} \qquad \mathbf{C}_2 \in \mathbb{R}^{n_2 \times 3}$$

Network architecture: Output from deconv branch

Two branch version

conv deconv fully connected set union

input

Encoder

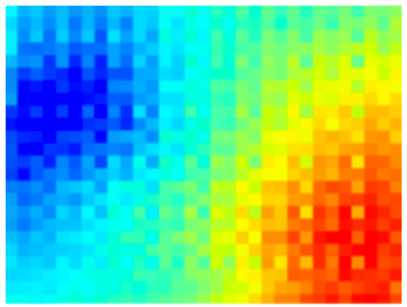
Predictor

point set

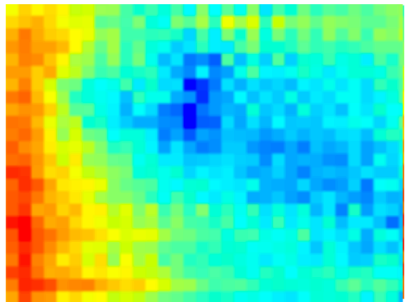
n_1=24*32=768 points

n_2=256 points

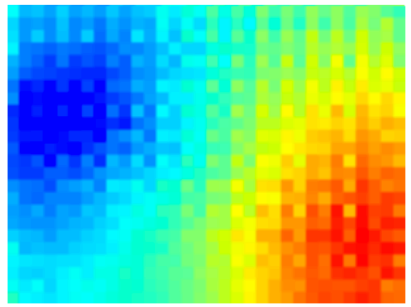
3-channel map of XYZ coordinates



x-channel



y-channel

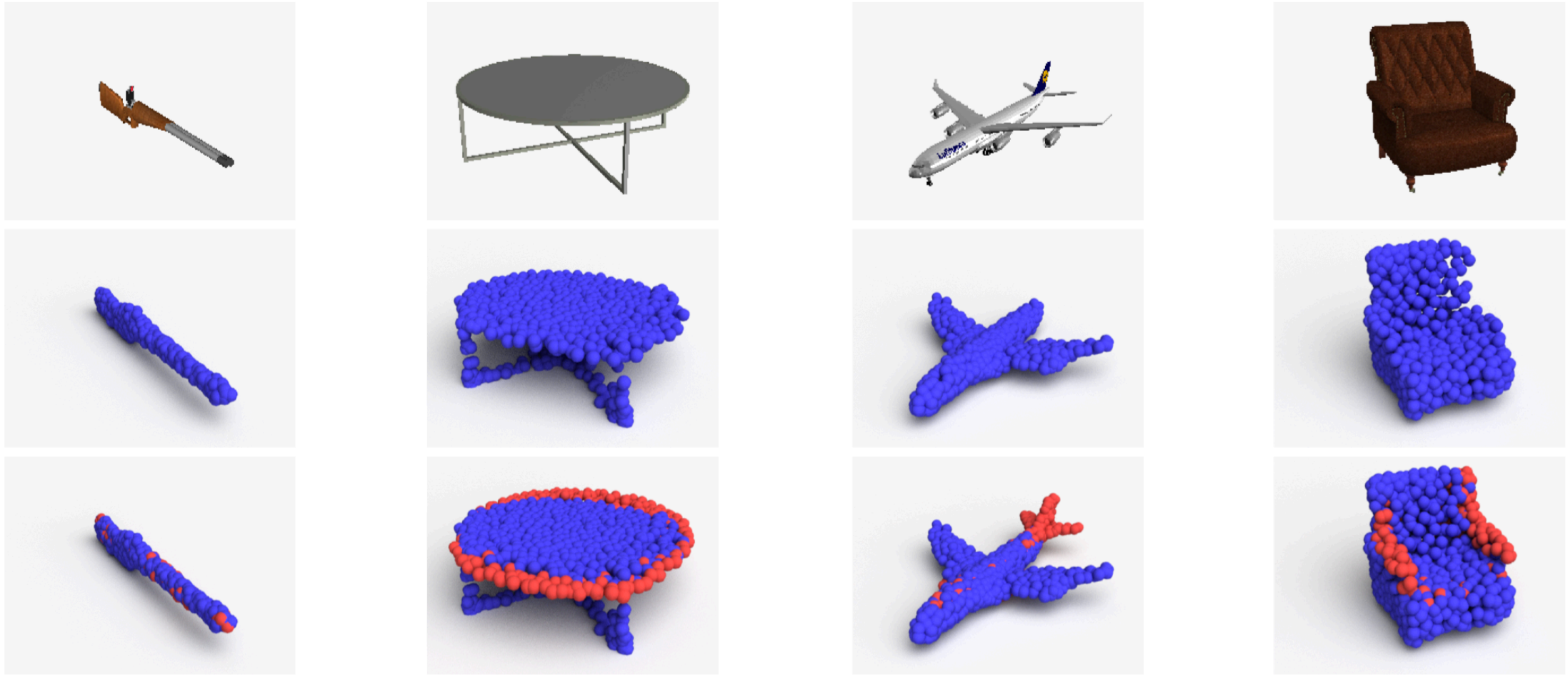


z-channel

Network architecture: The role of two branches

blue: deconv branch – **large, consistent, smooth** structures

red: fully-connected branch – **flexibly** reconstruct **intricate** structures



An end-to-end learning system

Deep Neural Network

Predicted set

$$\begin{Bmatrix} (x_1, y_1, z_1) \\ (x_2, y_2, z_2) \\ \dots \\ (x_n, y_n, z_n) \end{Bmatrix}$$

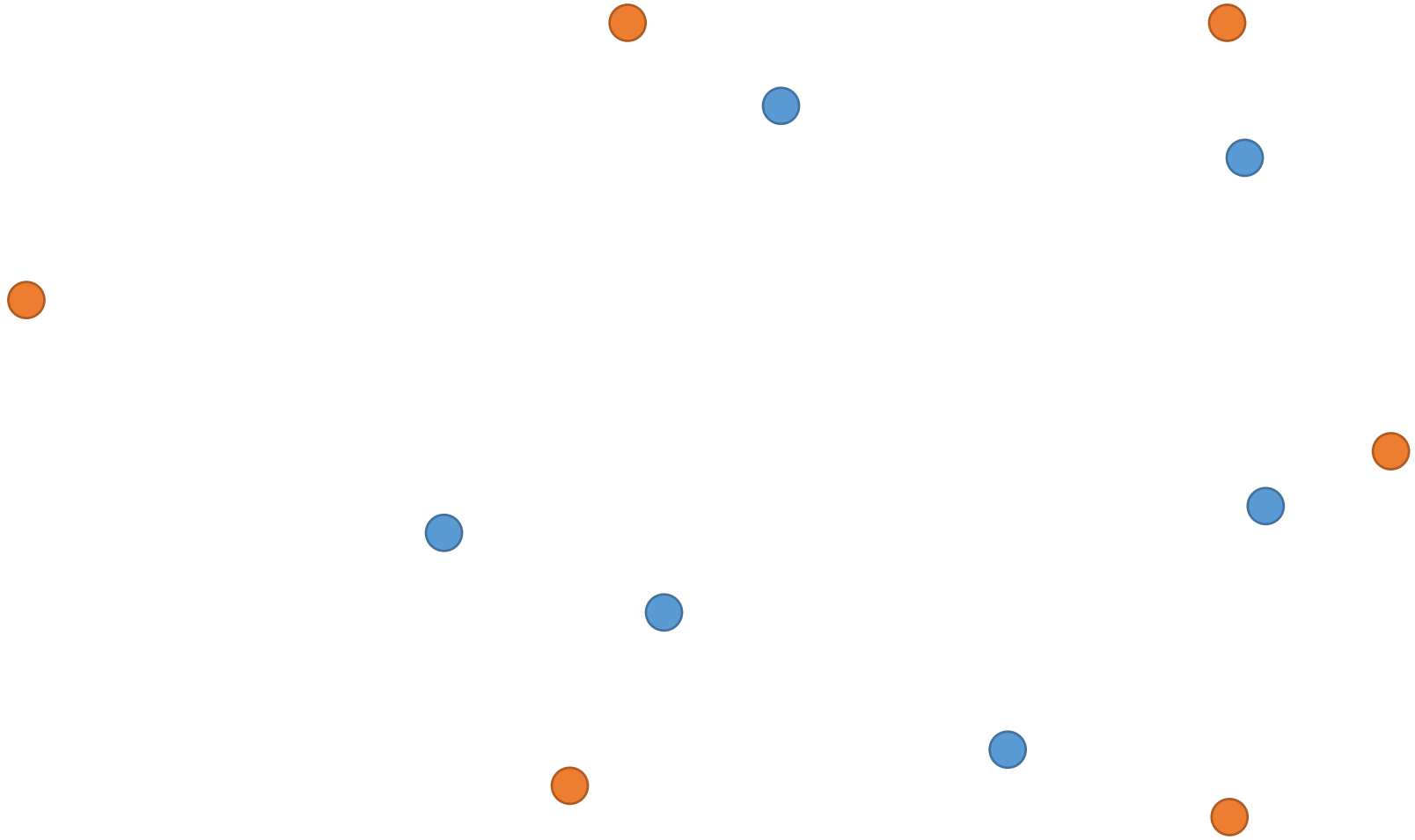
Point Set Loss

$$\begin{Bmatrix} (x'_1, y'_1, z'_1) \\ (x'_2, y'_2, z'_2) \\ \dots \\ (x'_n, y'_n, z'_n) \end{Bmatrix}$$

Groundtruth point cloud

Distance metrics between point sets

Given two sets of points, measure their discrepancy

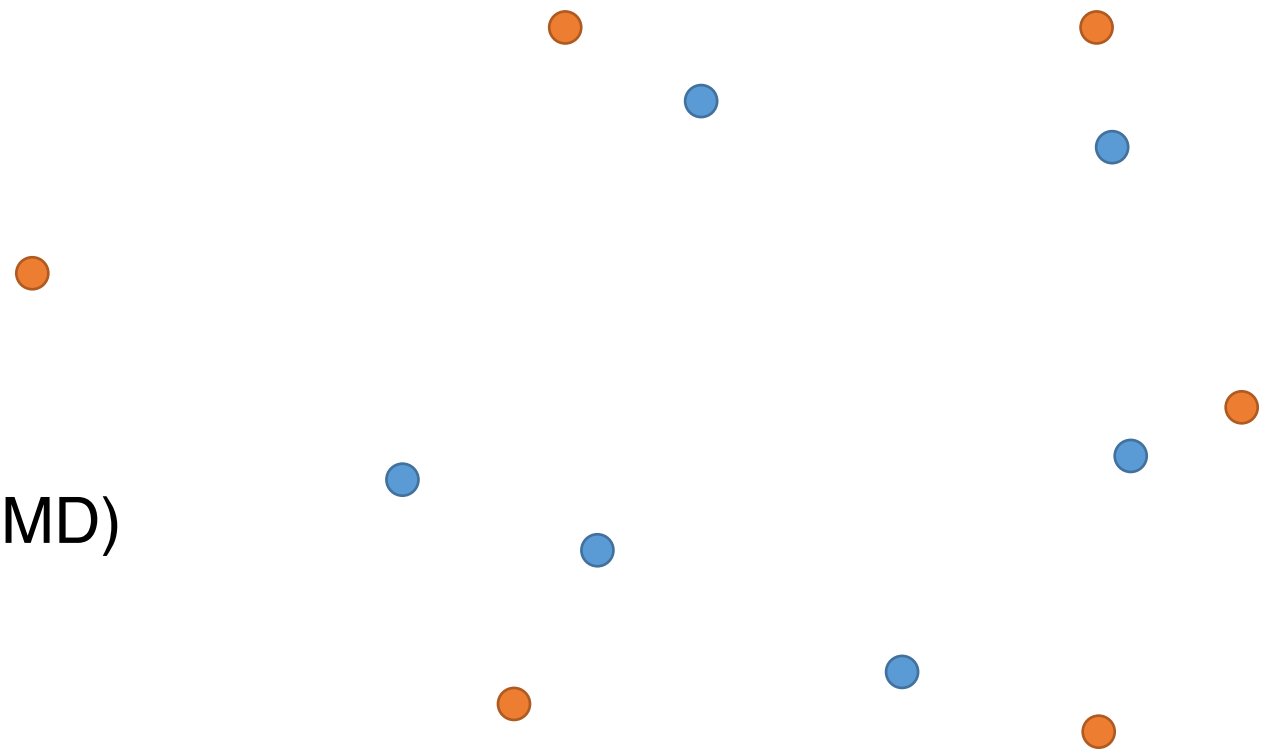


Common distance metrics

Worst case: Hausdorff distance (HD)

Average case: Chamfer distance (CD)

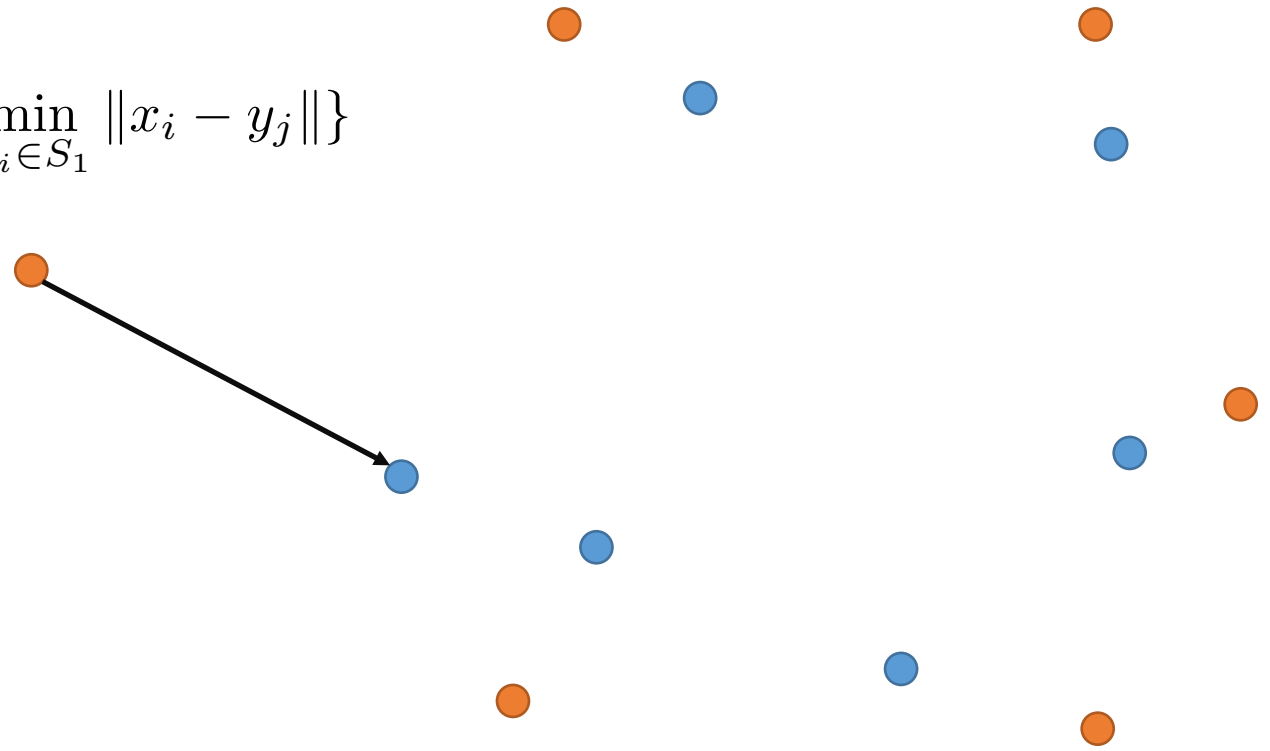
Optimal case: Earth Mover's distance (EMD)



Common distance metrics

Worst case: Hausdorff distance (HD)

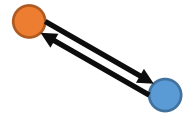
$$d_{\mathrm{HD}}(S_1, S_2) = \max\{\max_{x_i \in S_1} \min_{y_j \in S_2} \|x_i - y_j\|, \max_{y_j \in S_2} \min_{x_i \in S_1} \|x_i - y_j\|\}$$



A single farthest pair determines the distance.
In other words, ***not robust to outliers!***

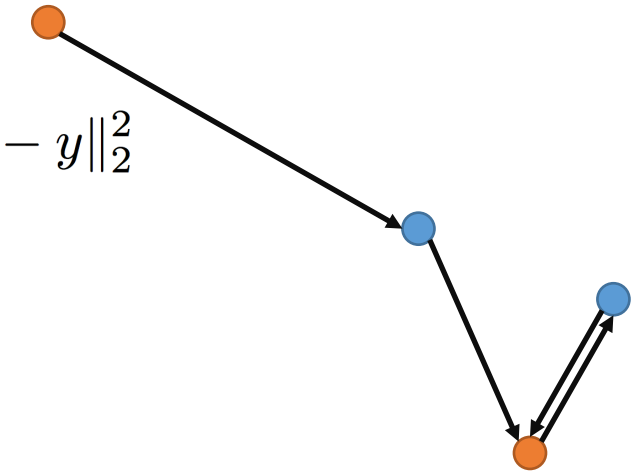
Common distance metrics

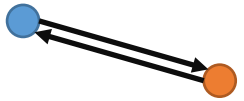
Worst case: Hausdorff distance (HD)



Average case: Chamfer distance (CD)

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$





Average all the nearest neighbor distance by nearest neighbors

Common distance metrics

Worst case: Hausdorff distance (HD)

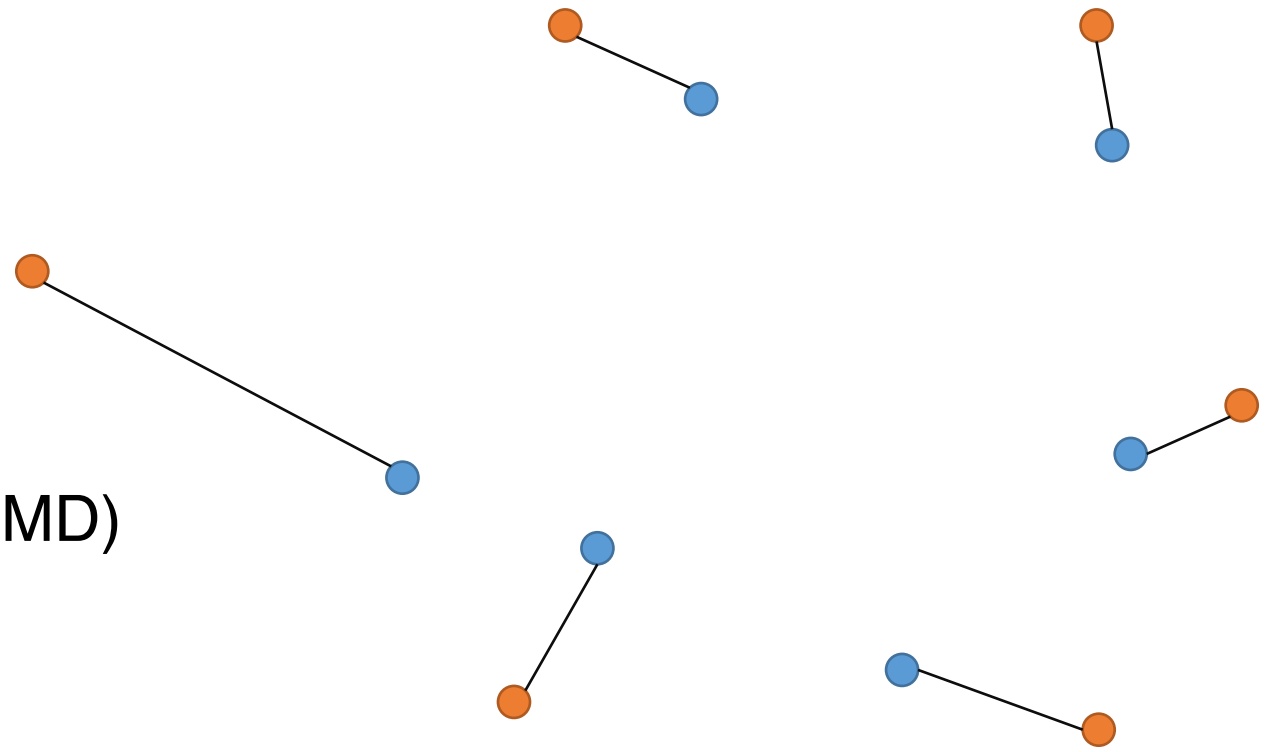
Average case: Chamfer distance (CD)

Optimal case: Earth Mover's distance (EMD)

$$d_{EMD}(S_1, S_2) = \min_{\phi: S_1 \to S_2} \sum_{x \in S_1} \|x - \phi(x)\|_2$$

where $\phi : S_1 \to S_2$ is a bijection.

Solves the optimal transportation (bipartite matching) problem!



Required properties of distance metrics

Geometric requirement

- Induces a nice shape space
- In other words, a good metric should reflect the natural shape differences

Computational requirement

- Defines a loss that is numerically easy to optimize

Required properties of distance metrics

Geometric requirement

- Induces a nice shape space
- In other words, a good metric should reflect the natural shape differences

Computational requirement

- Defines a loss that is numerically easy to optimize

How distance metric affects the learned geometry?

A fundamental issue: there is always uncertainty in prediction

By loss minimization, the network tends to predict a “**mean shape**” that averages out uncertainty in geometry

How distance metric affects the learned geometry?

A fundamental issue: there is always uncertainty in prediction, due to

- limited network ability
- Insufficient training data
- inherent ambiguity of groundtruth for 2D-3D dimension lifting
- etc.

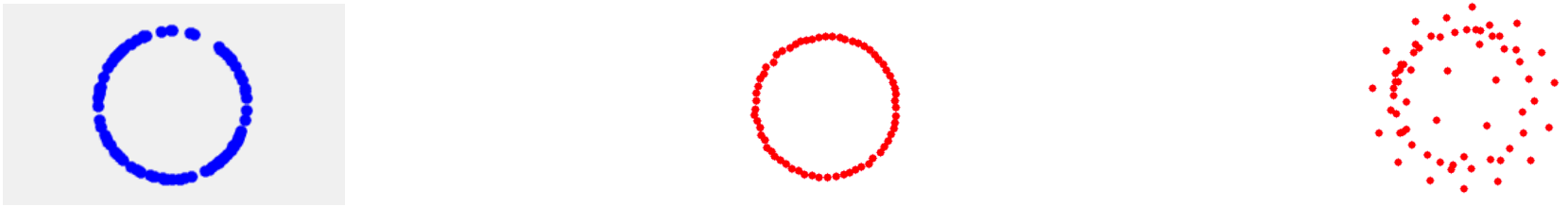
By loss minimization, the network tends to predict a "**mean shape**" that averages out uncertainty in geometry

Mean shapes are affected by distance metric

The mean shape carries characteristics of the distance metric

$$\bar{x} = \underset{x}{\operatorname{argmin}}\ \mathbb{E}_{s\sim\mathbb{S}}[d(x, s)]$$

continuous hidden variable (radius)



Input

EMD mean

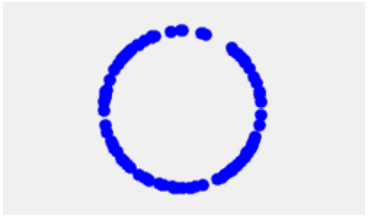
Chamfer mean

Mean shapes from distance metrics

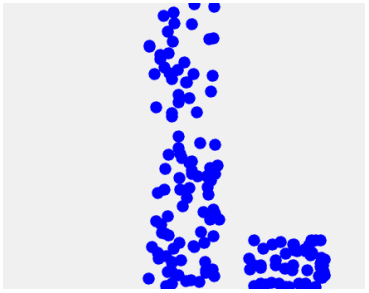
The mean shape carries characteristics of the distance metric

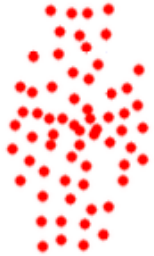
$$\bar{x} = \underset{x}{\operatorname{argmin}}\ \mathbb{E}_{s\sim\mathbb{S}}[d(x,s)]$$

continuous hidden variable (radius)



discrete hidden variable (add-on location)





Input

EMD mean

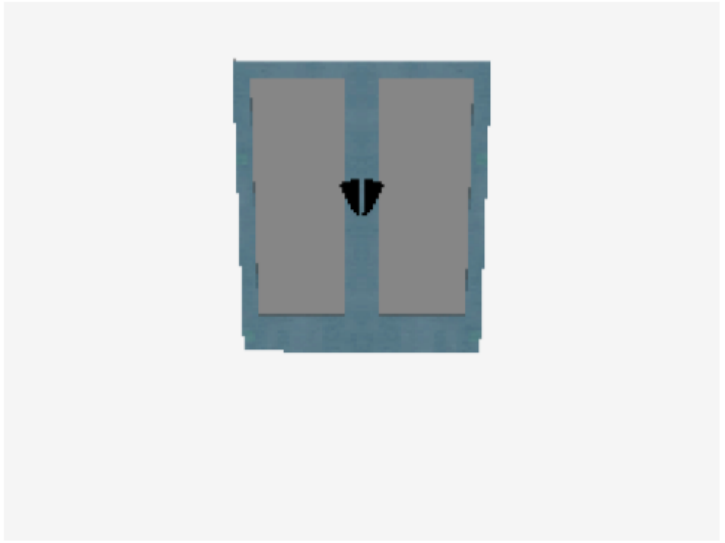
Chamfer mean

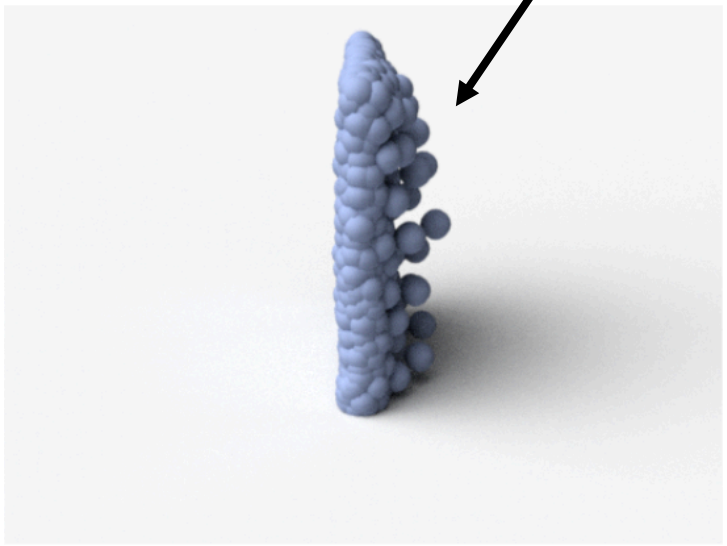
Comparison of predictions by CD versus EMD

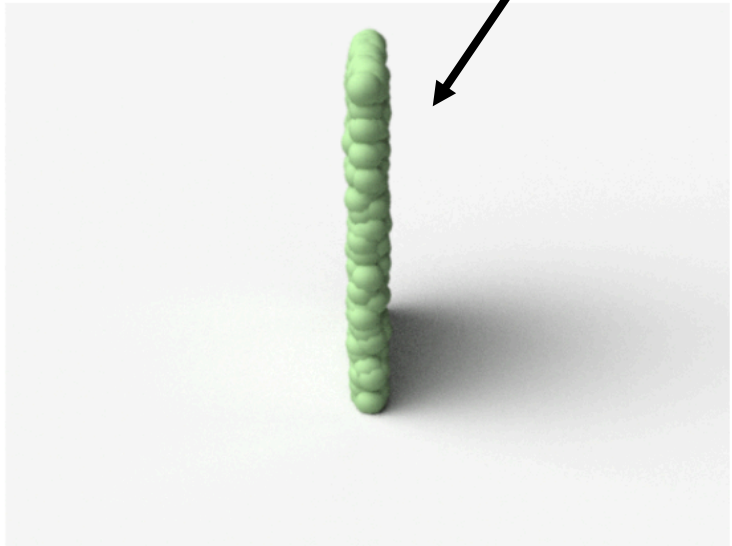
Input

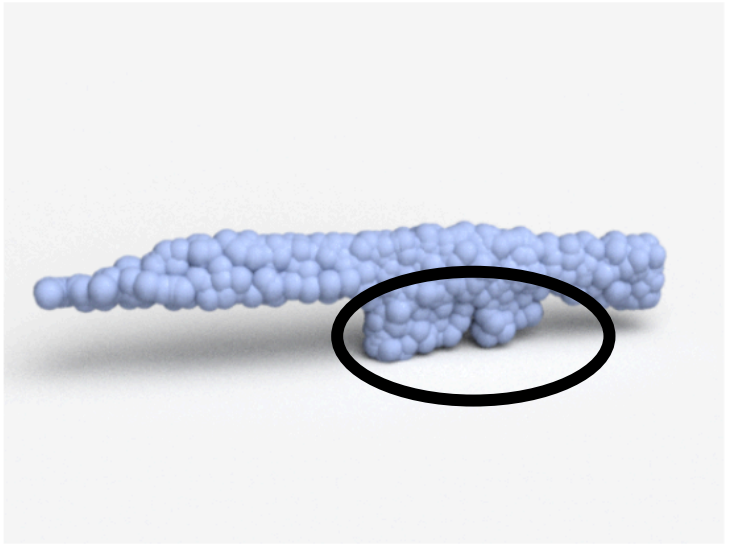
Chamfer

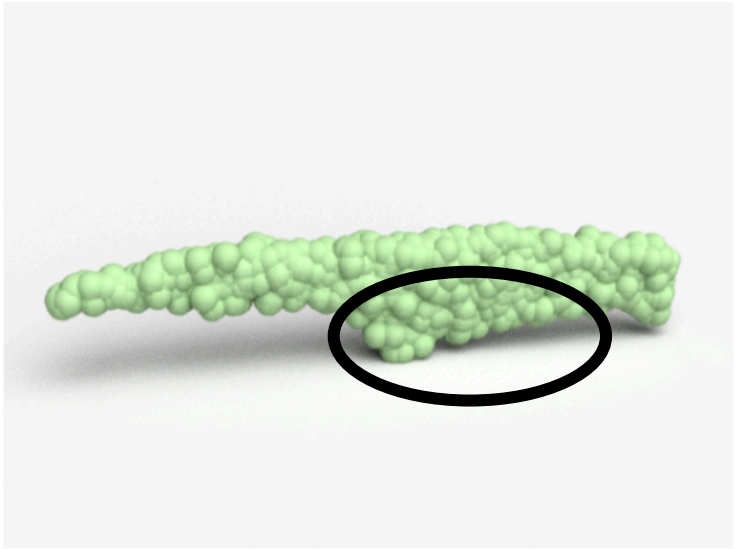
EMD











Lower prediction uncertainty, better mean shapes

Can we reduce prediction uncertainty by

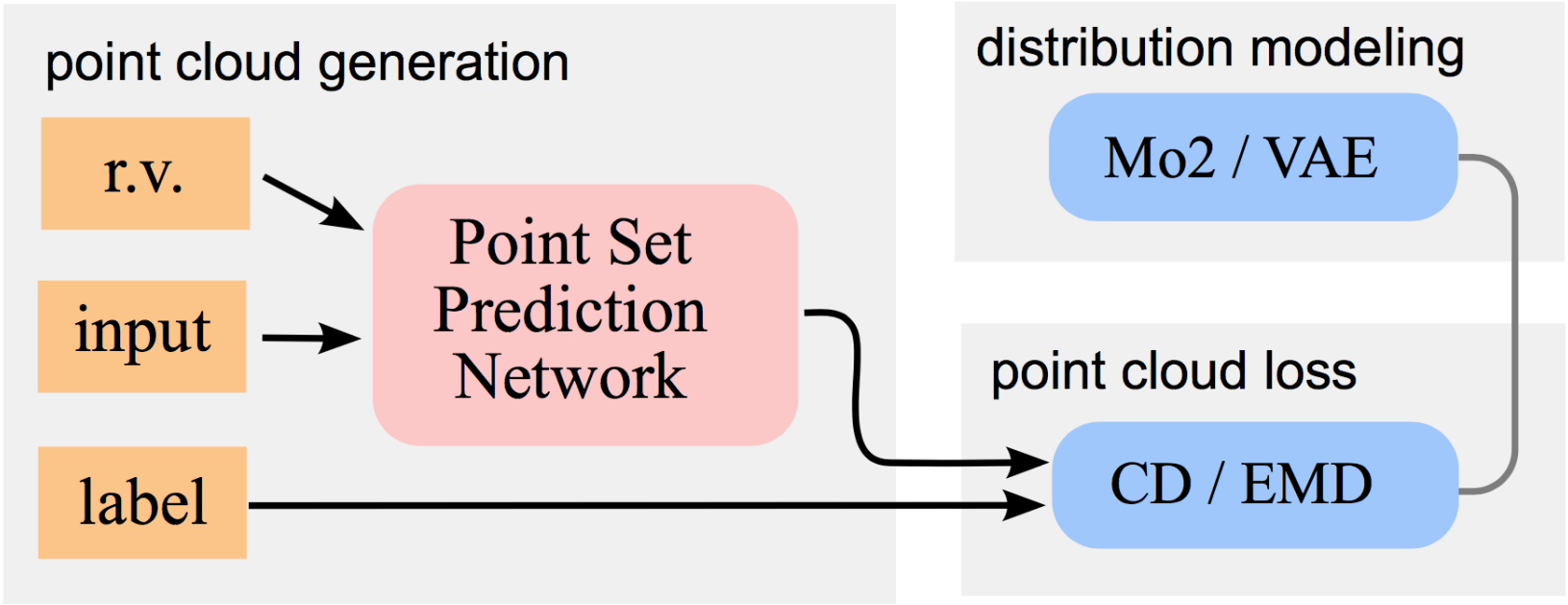
factoring out the inherent ambiguity of groundtruth?

Input

Possible observations from a novel viewpoint

Predict multiple candidates

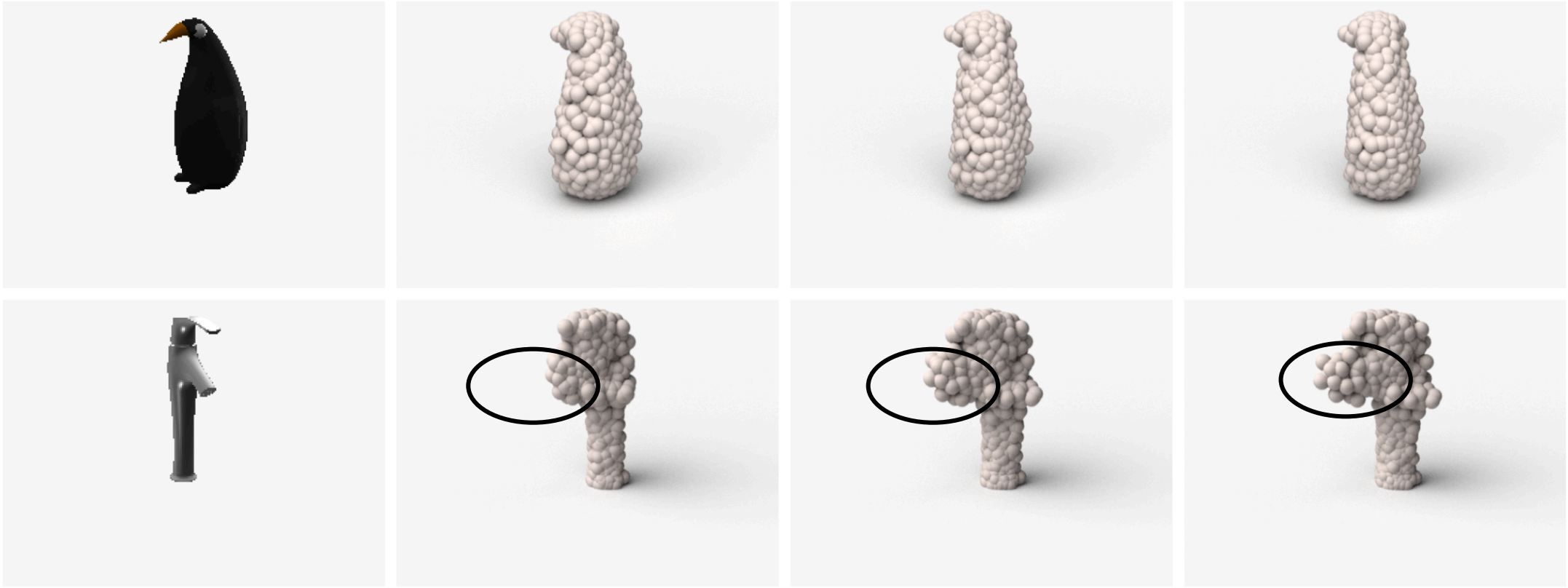
Build a conditional shape sampler $\mathbb{G}(I, r)$



- r is a random variable to perturb input
- Can navigate the groundtruth distribution

Can be trained by conditional VAE or our MoN loss

Multiple plausible 3D shape predictions



45 deg

side view

Required properties of distance metrics

Geometric requirement

- Induces a nice shape space
- In other words, a nice metric should reflect the natural shape difference

Computational requirement

- Defines a loss function that is numerically easy to optimize

Computational requirement of metrics

To be used as a loss function, the metric has to be

- **Differentiable** with respect to point locations
- **Efficient** to compute

Computational requirement of metrics

- **Differentiable** with respect to point location

Chamfer distance

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$

Earth Mover's distance

$$d_{EMD}(S_1, S_2) = \min_{\phi: S_1 \to S_2} \sum_{x \in S_1} \|x - \phi(x)\|_2 \quad \text{where } \phi : S_1 \to S_2 \text{ is a bijection.}$$

- Simple function of coordinates
- In general positions, the correspondence is unique
- With infinitesimal movement, the correspondence does not change

Conclusion: differentiable almost everywhere

Computational requirement of metrics

- **Efficient** to compute

  Chamfer distance: trivially parallelizable on CUDA

  Earth Mover's distance:

  - Use coarse-to-fine approximation algorithm (Bertsekas, 1985)
  - Quite good approximation ratio
  - Parallelizable

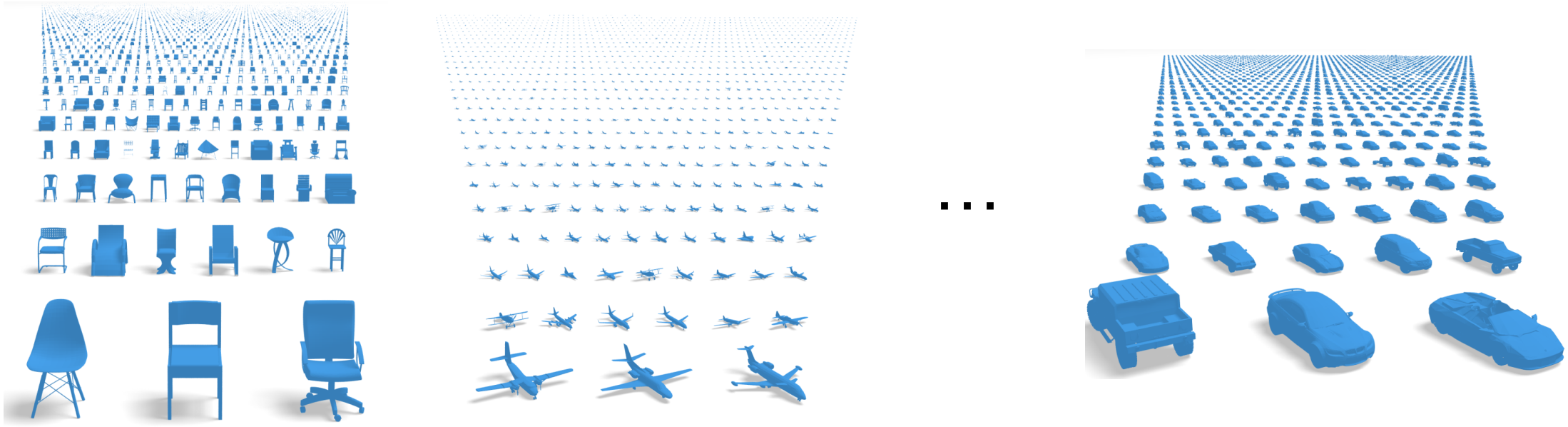
Training

Implemented in TensorFlow (python)

Converge in ~2 days (the two branch version)

Trained on 4 GPUs in parallel

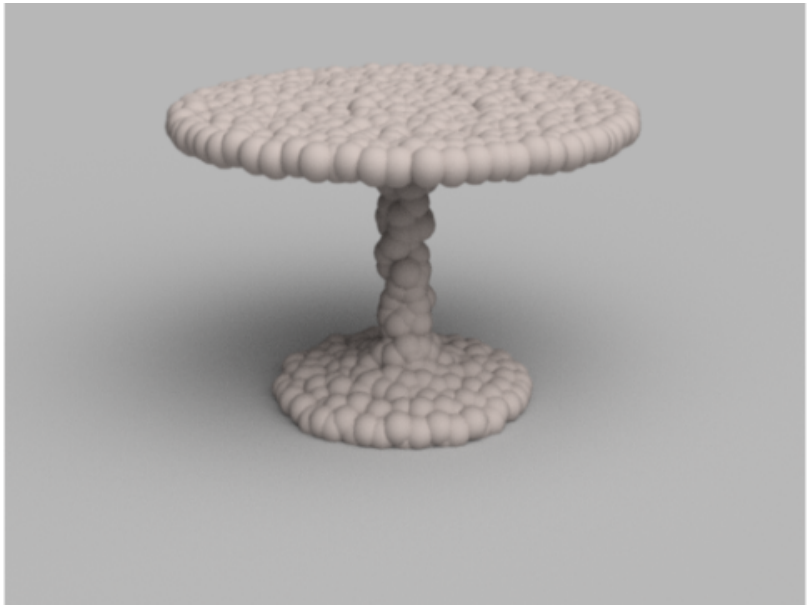
Training data rendered from **220K** shapes in ShapeNet, covering **~2K** categories

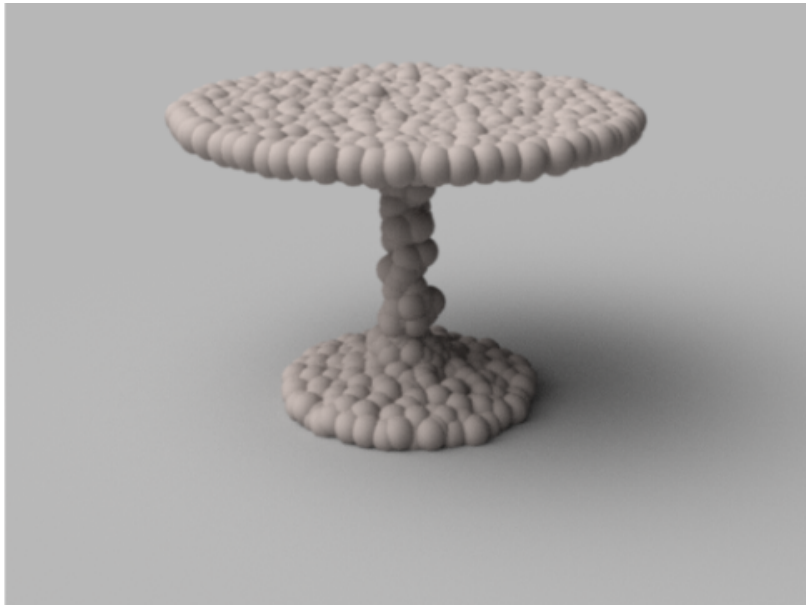


More Results

More visual results

Good symmetry





View 1

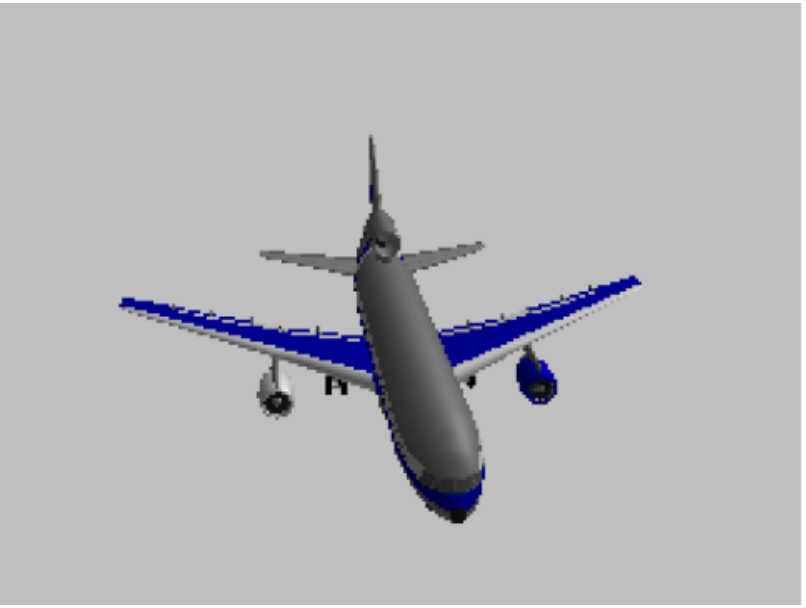
View 2

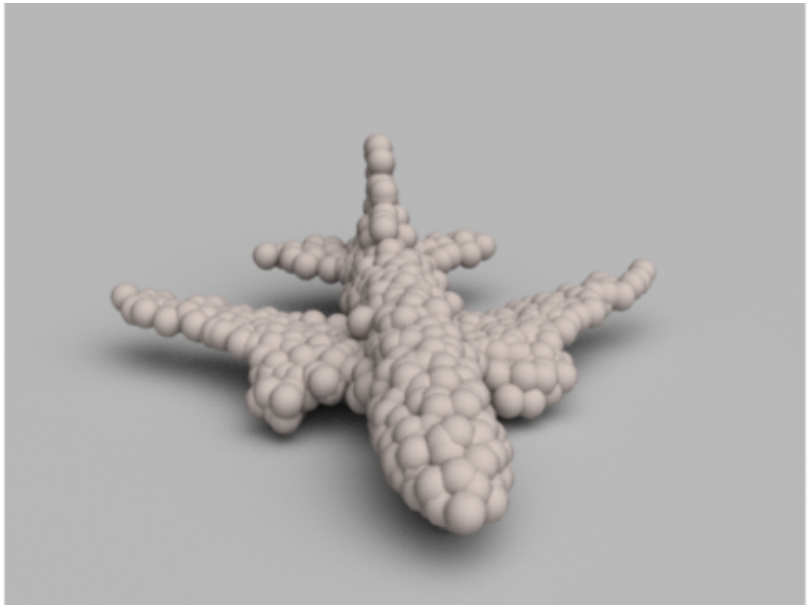
Input

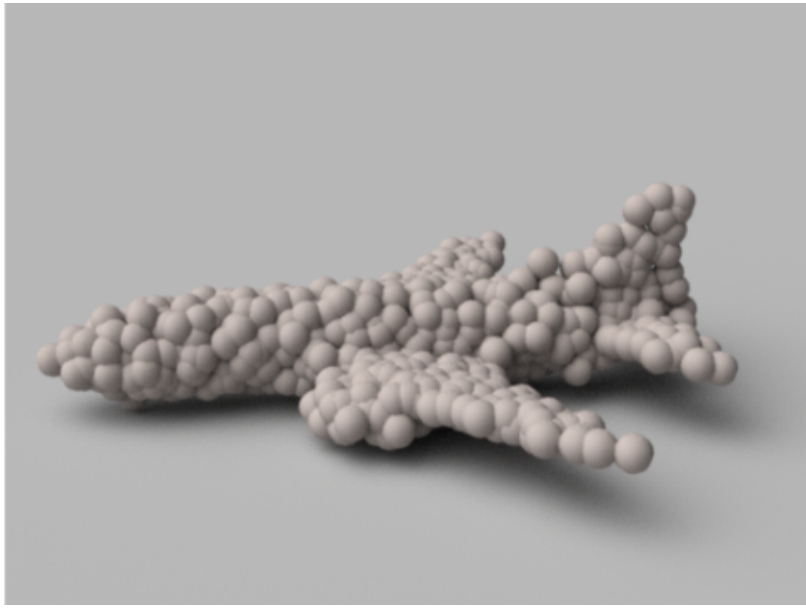
Prediction

More visual results

Good details







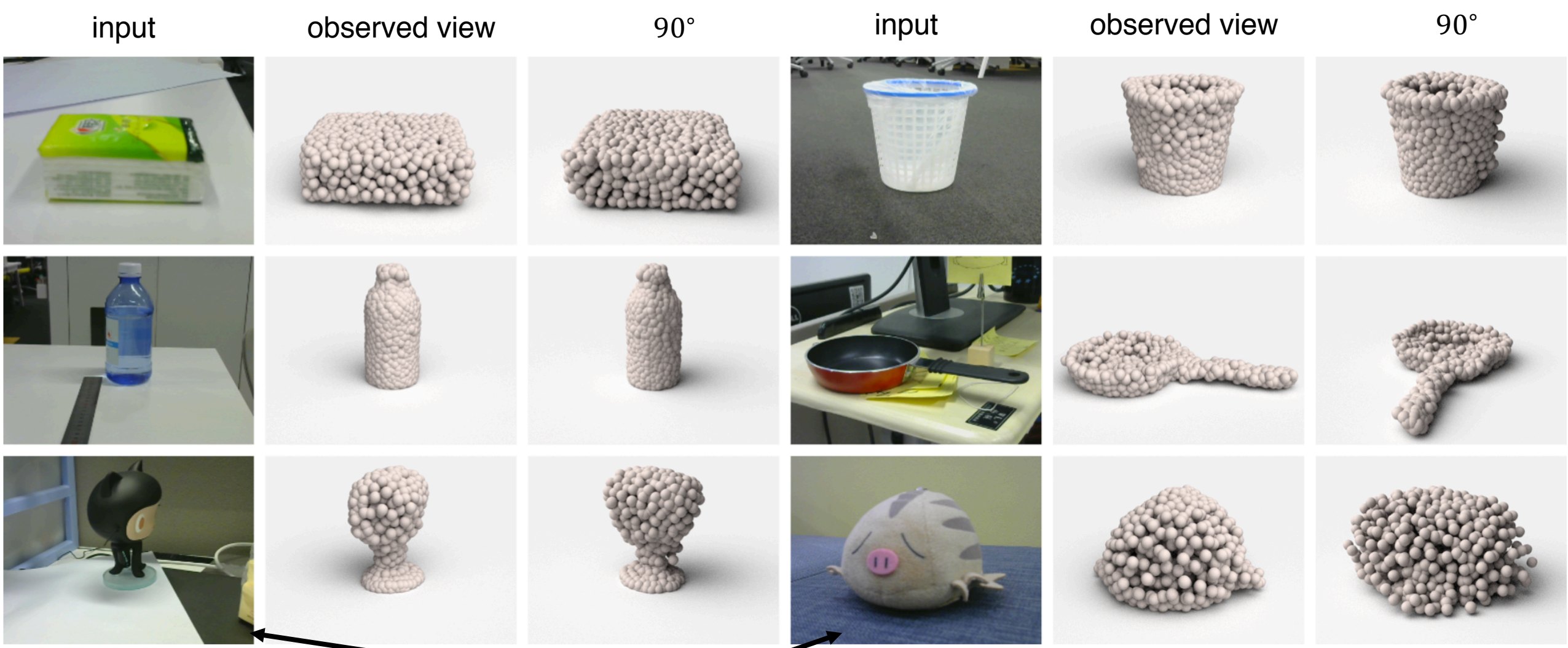
View 1

View 2

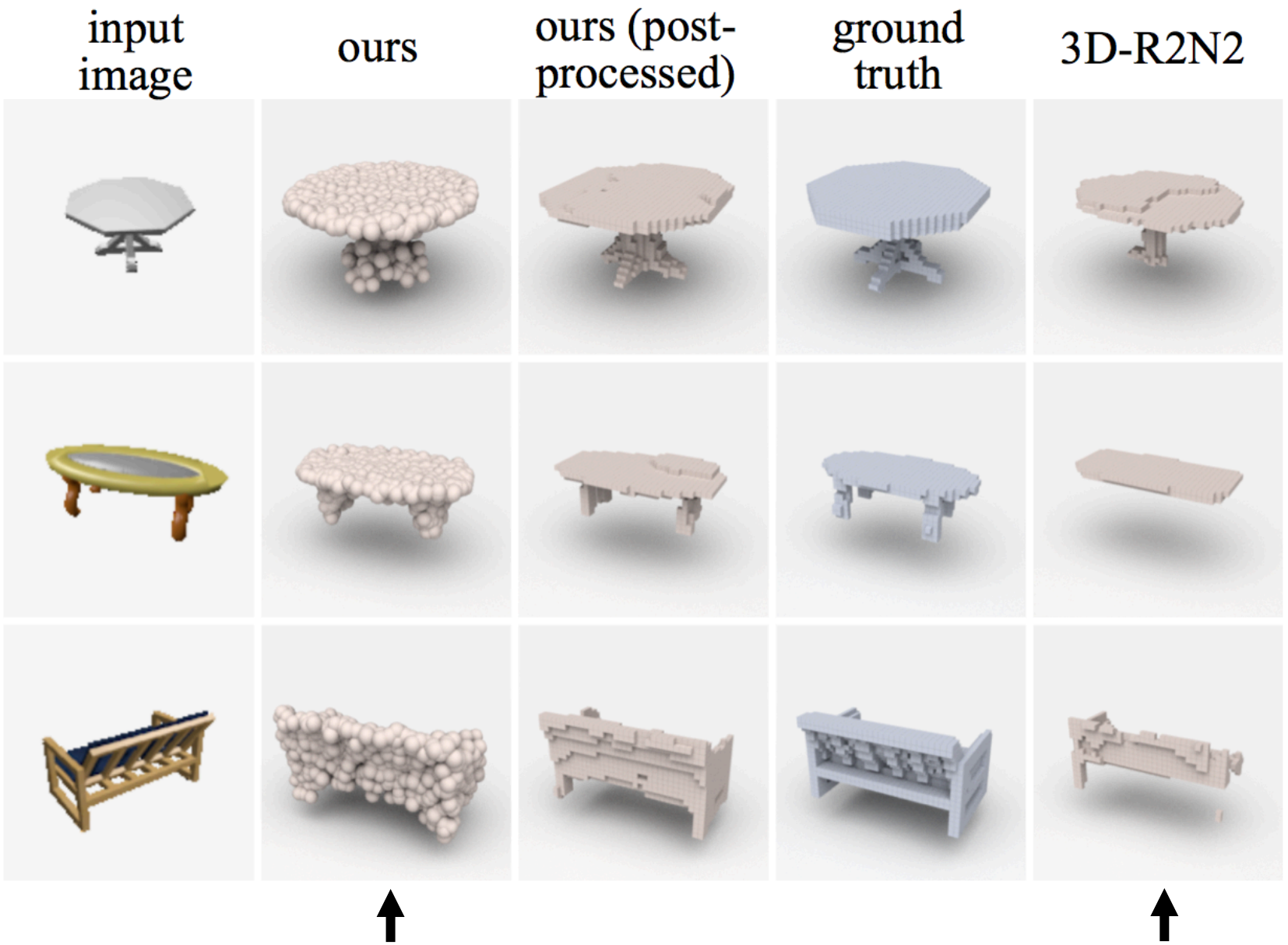
Input

Prediction

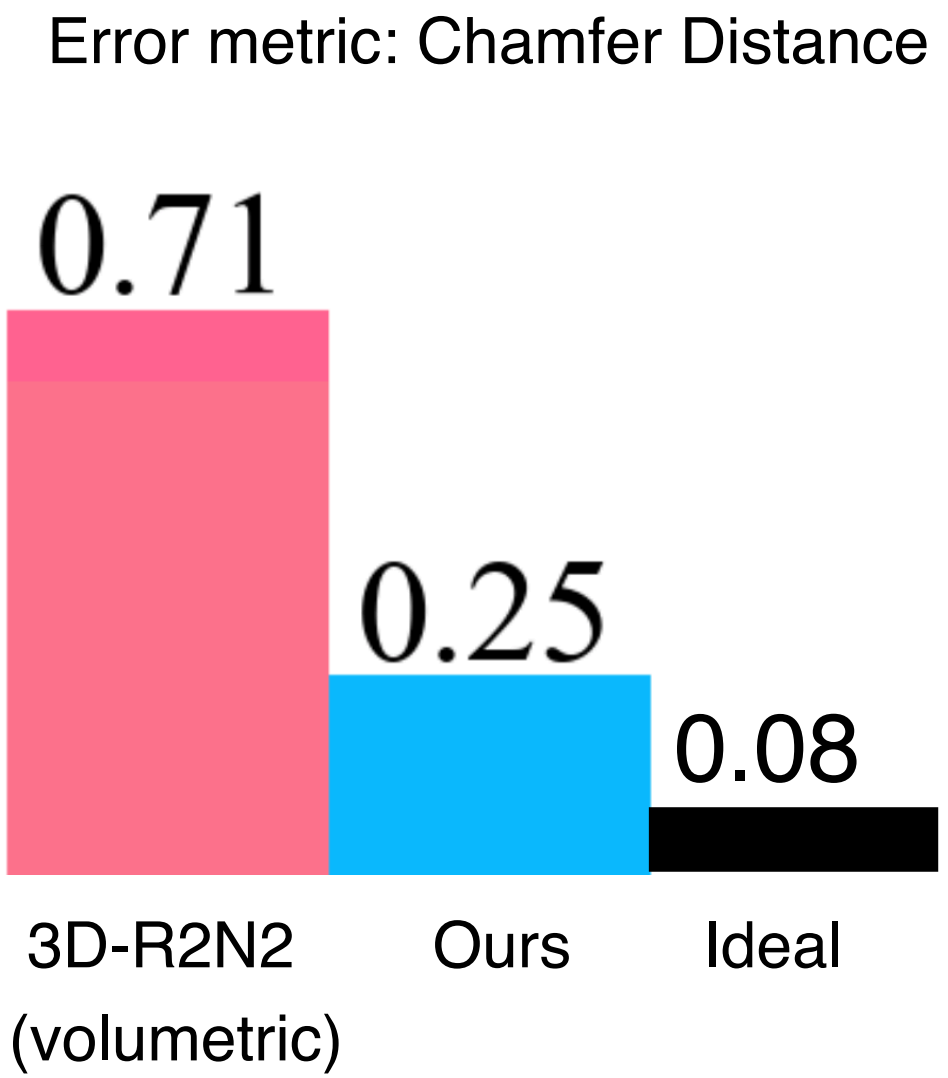
Real-world results



Comparison with state-of-the-art (volumetric)



Comparison with state-of-the-art (volumetric)



Shape completion from depth map

How about learning to predict geometric forms?

Candidates:

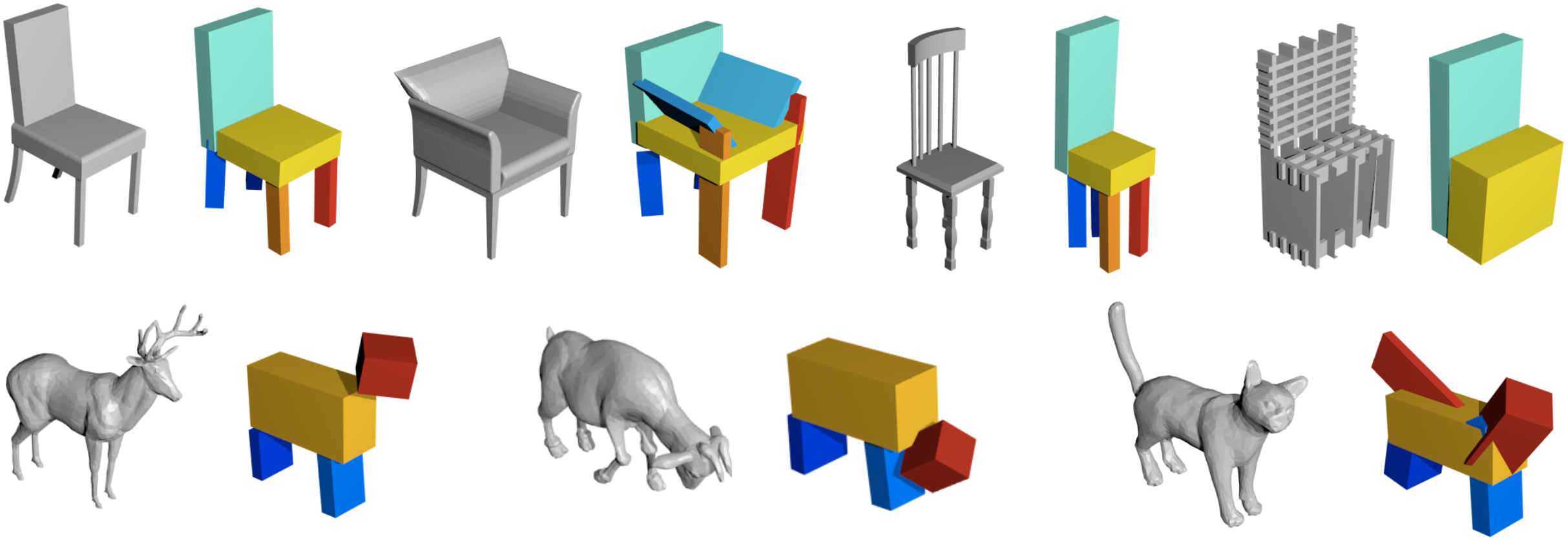
Rasterized form (regular grids)

- multi-view images
- depth map
- volumetric

Geometric form (irregular)

- polygonal mesh
- **point cloud** ★
- **primitive-based CAD models**

Primitive-based assembly

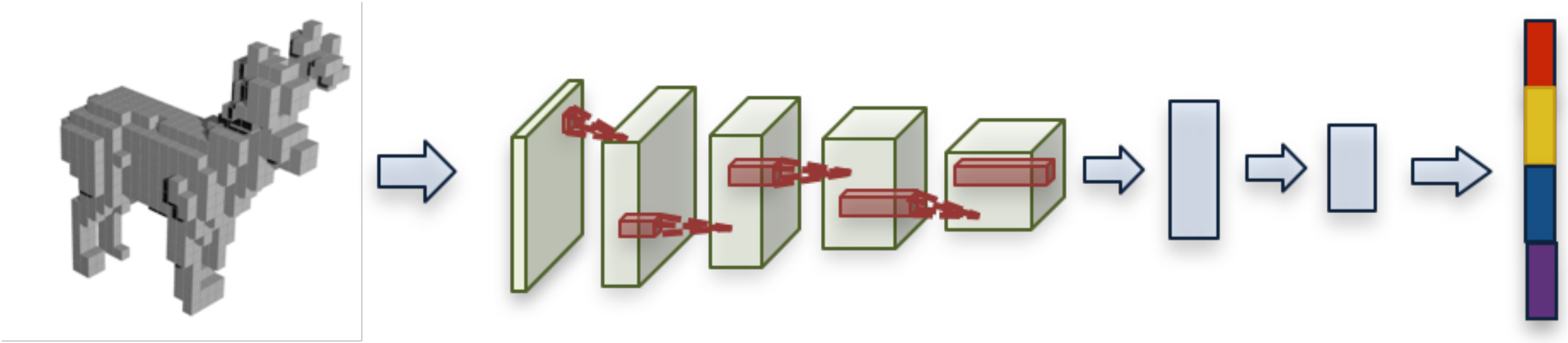


We **learn** to predict a corresponding shape composed by primitives.
It allows us to predict **consistent** compositions across objects.

Unsupervised parsing

Each point is colored according to the assigned primitive

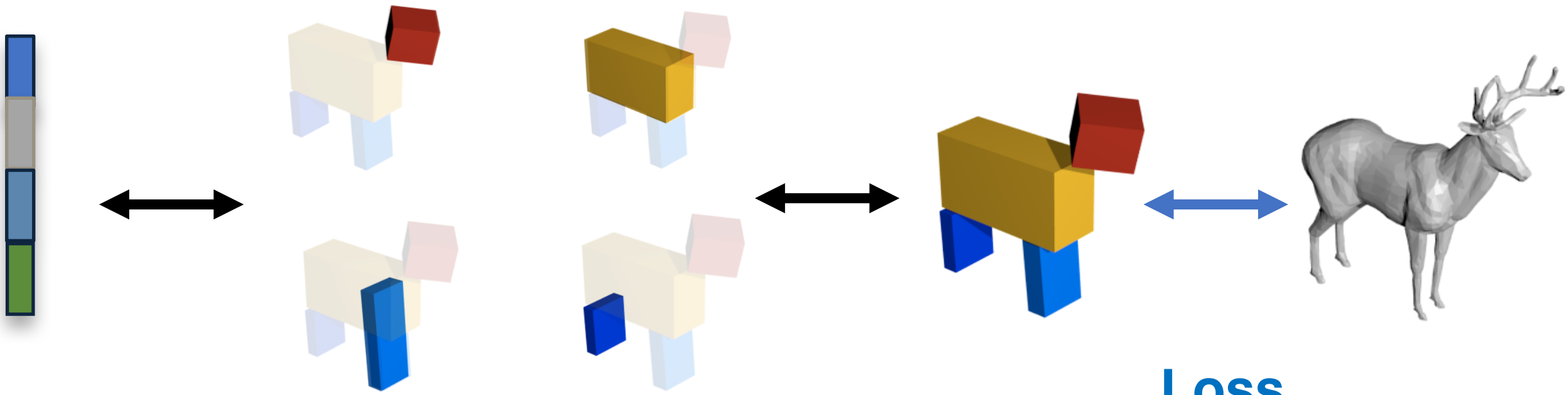
Approach – predict a high-dimensional point set



Primitive parameters as a point: size, rotation, translation of M cuboids.

Variable number of parts? We predict “primitive existence probability”

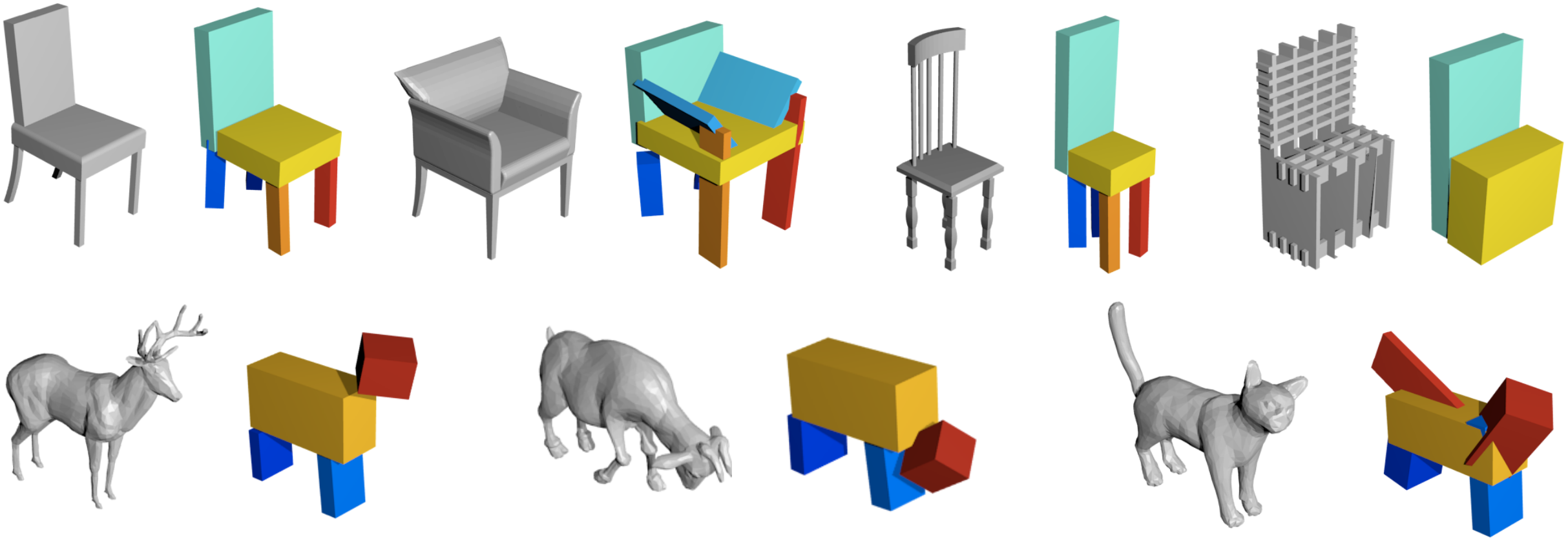
Loss function



Loss

Chamfer distance!

Consistent primitive configurations



Primitive locations are **consistent** due to
the **smoothness** of primitive prediction network

Unsupervised parsing

Method	[31] (initial)	[31] (refined)	Ours
Accuracy	78.6	84.8	89.0

Mean accuracy (face area) on Shape COSEG chairs.

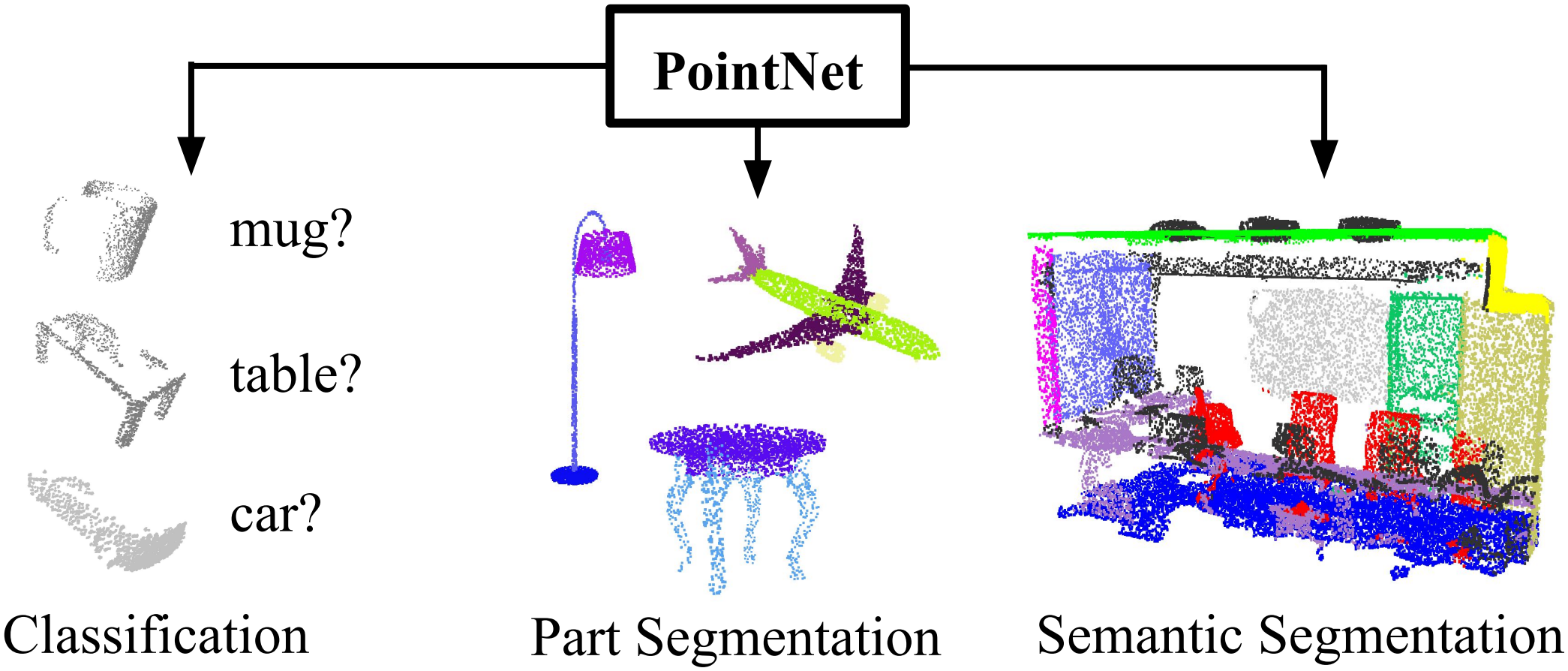
Outline

Motivation

3D point cloud / CAD model reconstruction

3D point cloud analysis

Deep Learning on Point Sets



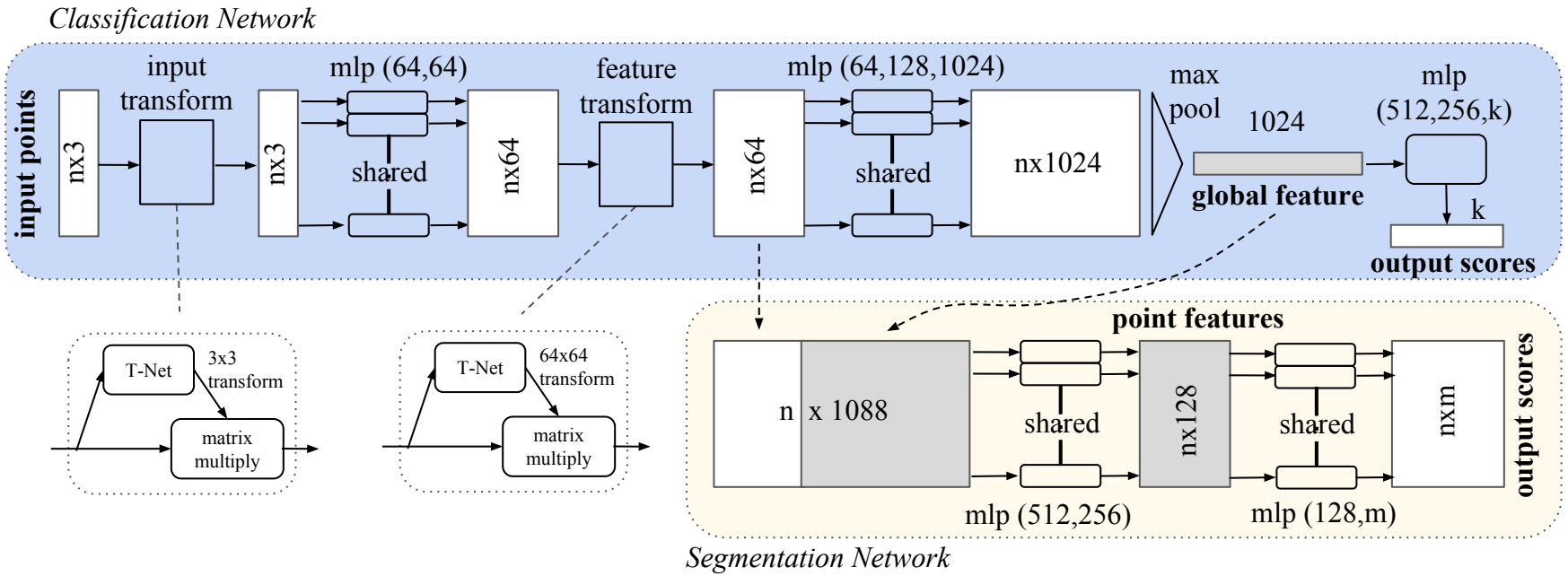
PointNet: deep net for unordered point set input

Idea 1: Sort ✗

Idea 2: RNN ✗

Idea 3: Use symmetry function ✓

E.g. max, sum, weighted sum, L-norm, histogram, polynomial etc.



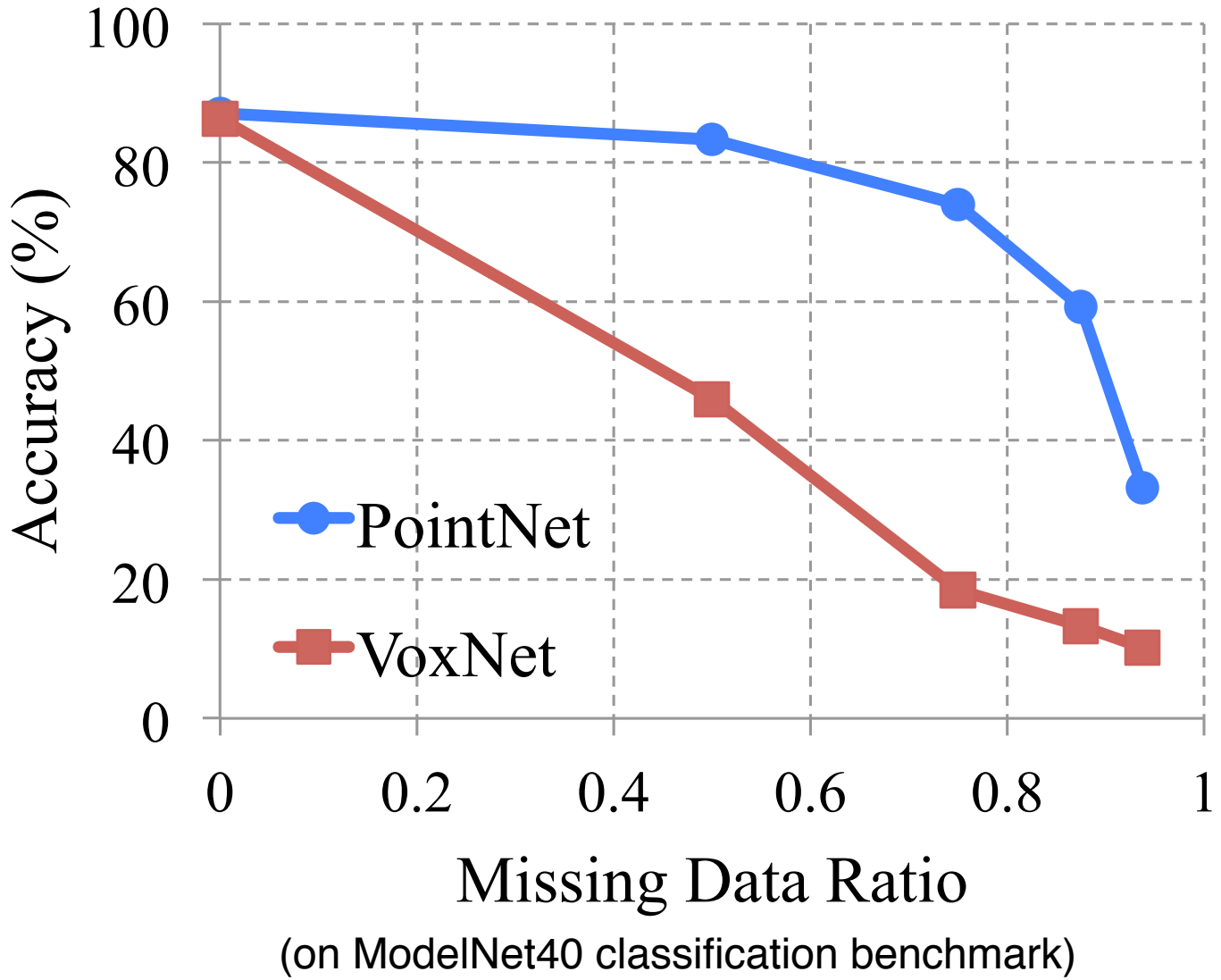
Universal approximation to continuous set functions

Theorem 1. *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* $g(x_1,\ldots,x_n) = \gamma \circ \boldsymbol{MAX}$, *such that for any* $S \in \mathcal{X}$,

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

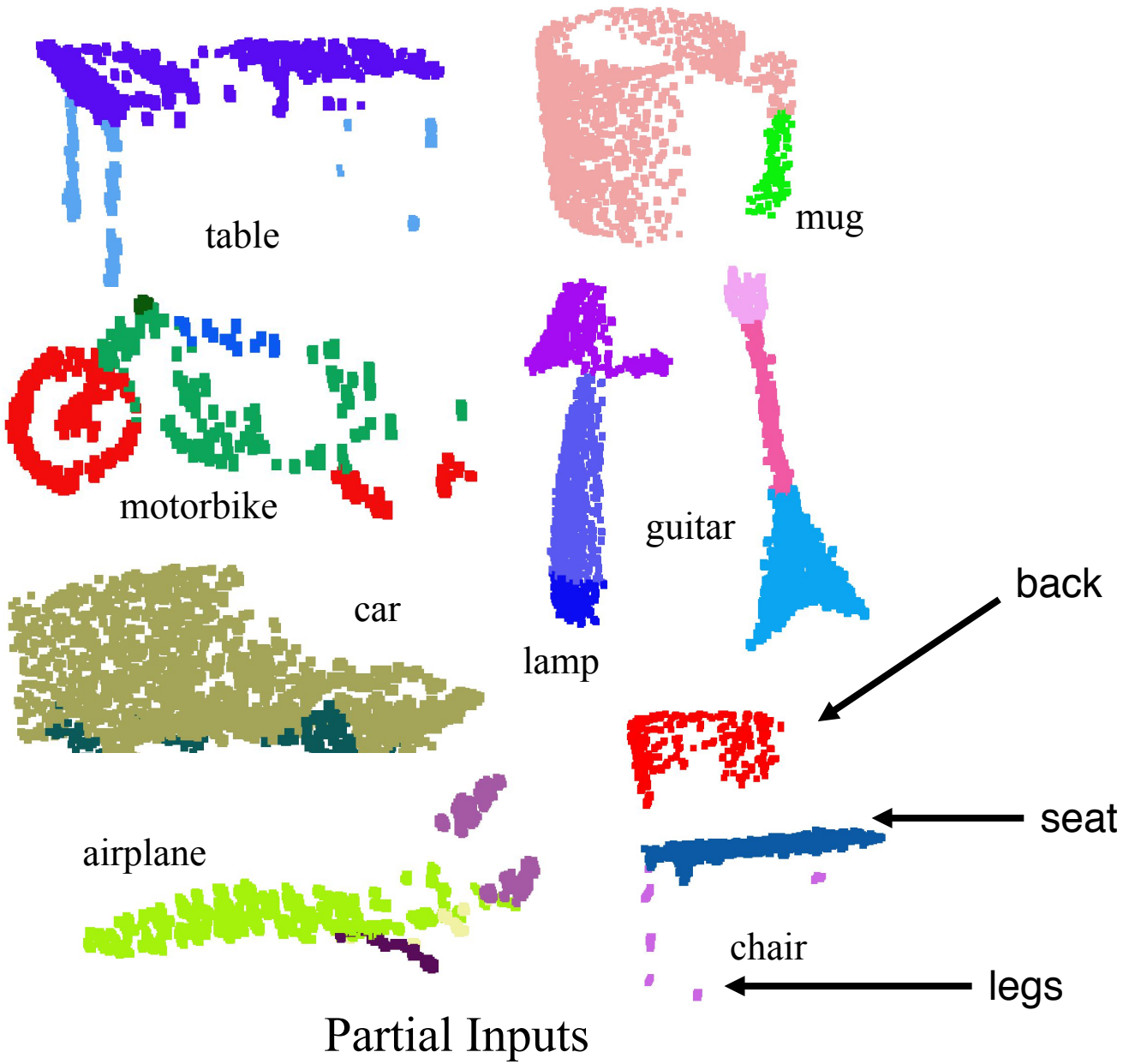
where $x_1,\ldots,x_n$ *is the full list of elements in* S *ordered arbitrarily,* γ *is a continuous function, and* $\boldsymbol{MAX}$ *is a vector max operator that takes* n *vectors as input and returns a new vector of the element-wise maximum.*

Robustness to data corruption

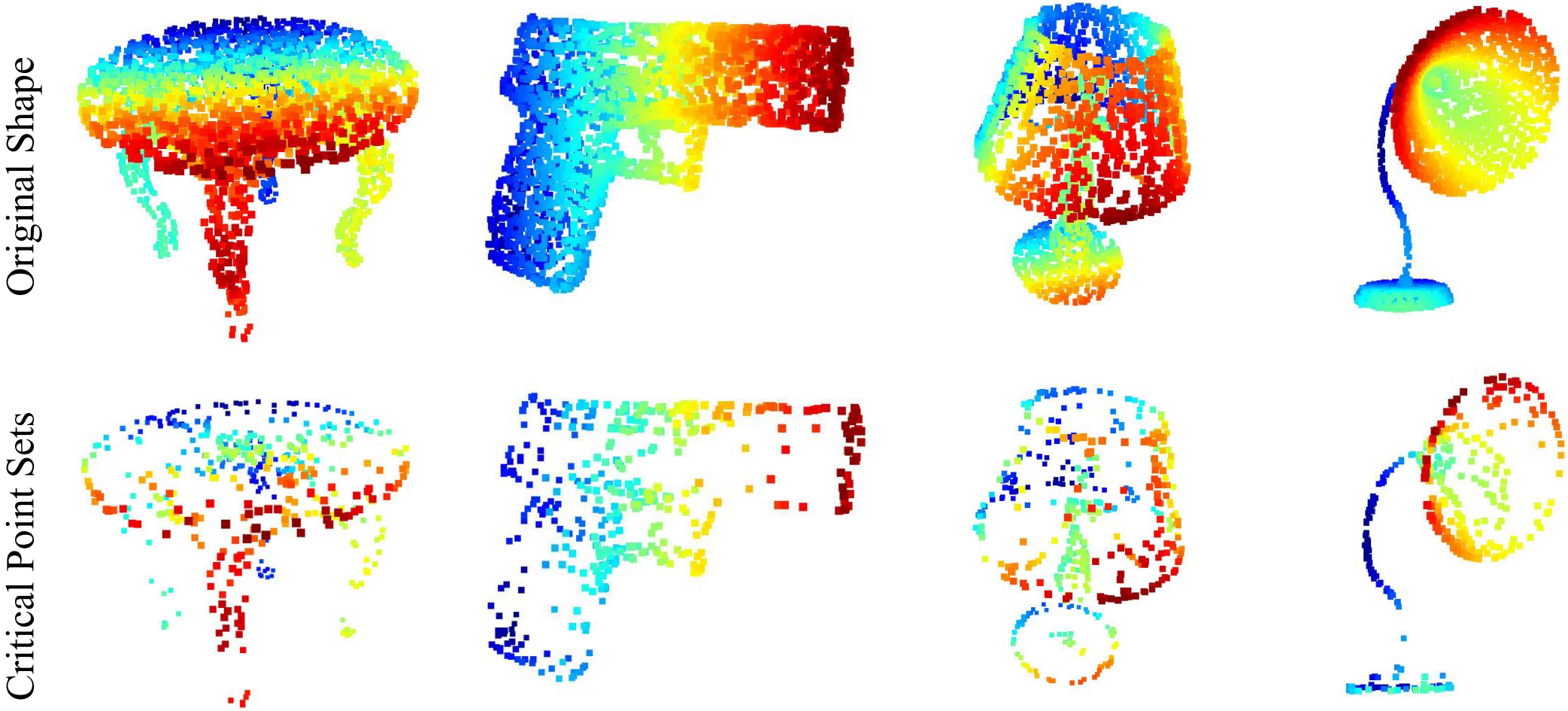


(on ModelNet40 classification benchmark)

Partial object part segmentation



Visualization of what are learned, by reconstruction



A compact summarization of the input set. Saliency!

To sum up

- We explore geometric representations as input / output of networks
- A space rich of open problems and opportunities

- Papers:

  Hao Su*, Haoqiang Fan*, Leonidas Guibas, A Point Set Generation Network for 3D Object Reconstruction from a Single Image, arxiv

  Hao Su*, Charles Qi*, Kaichun Mo, Leonidas Guibas, PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, arxiv

  Shubham Tulsiani, Hao Su, Leonidas Guibas, Alexei Efros, Jitendra Malik, Learning Shape Abstractions by Assembling Volumetric Primitives, arxiv

- Codes will be released soon!

Thank You!