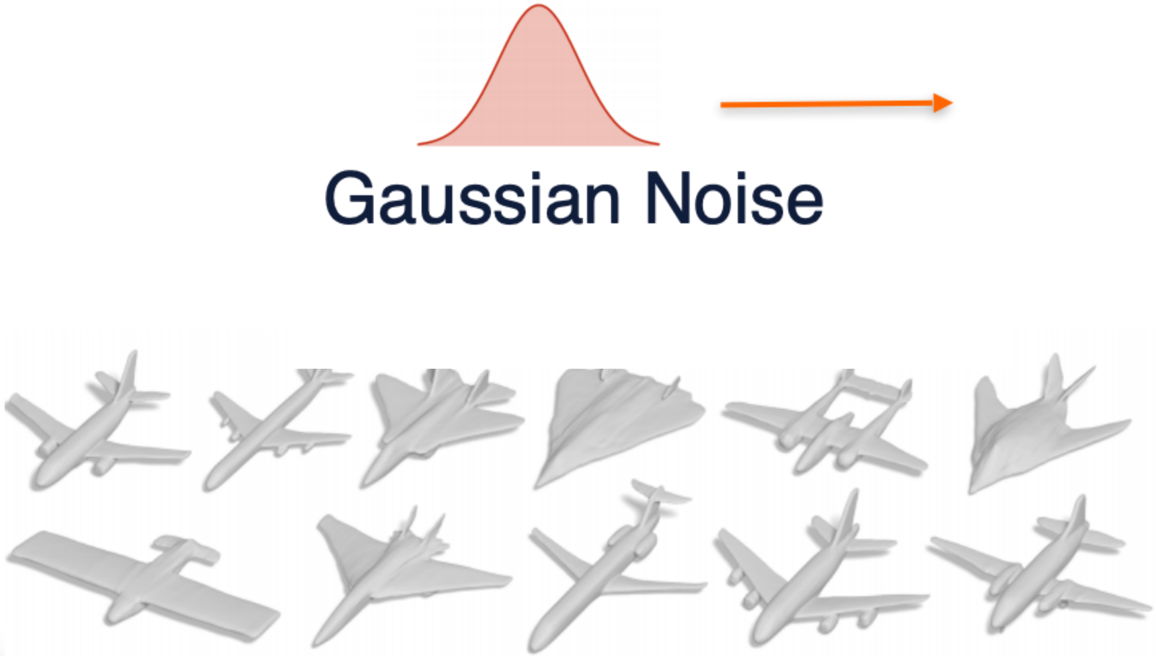
Learning 3D Representations and Generative Models

He Wang
Stanford University

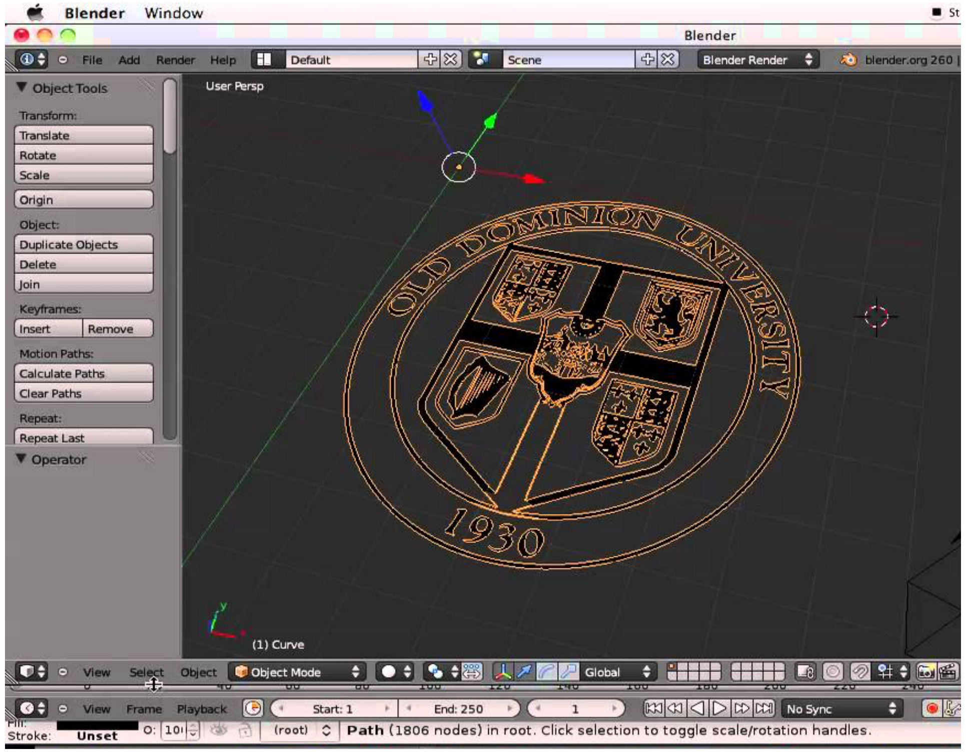


Leonidas Guibas Laboratory

Geometric Computing

Classic Methods for Generating 3D Data

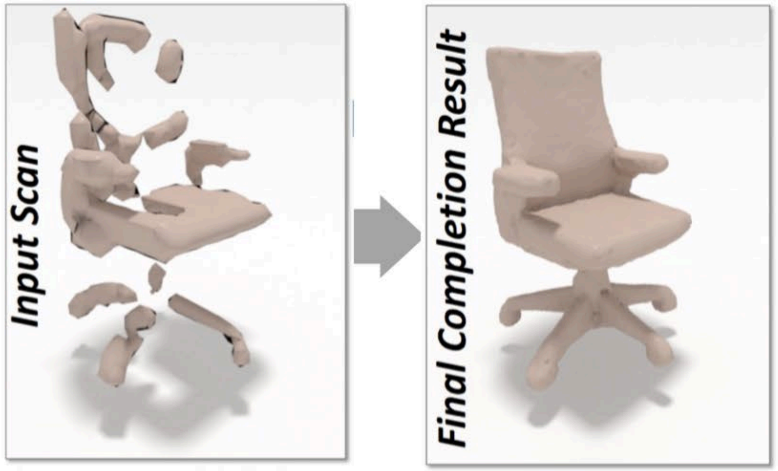
Creating CAD Model



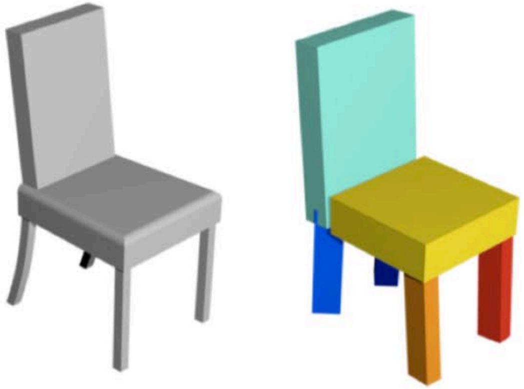
Scanning 3D Model

3D Deep Generative Models

Monocular
3D reconstruction



Shape completion



Shape modeling

Generative Model (**unconditional**)

Given training data, generate new samples from the same distribution:

Training data ~ $p_{data}(x)$

Generated samples~ $p_{model}(x)$

Objective: learn a $p_{model}(x)$ that matches $p_{data}(x)$.

Conditional Generative Model

- Data: (x, y) where x is a **condition** and y is the corresponding **content.**

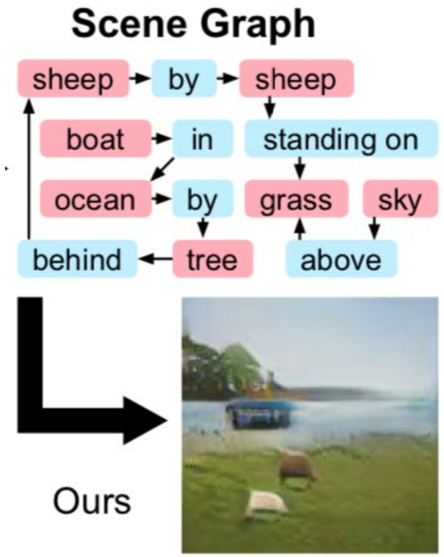
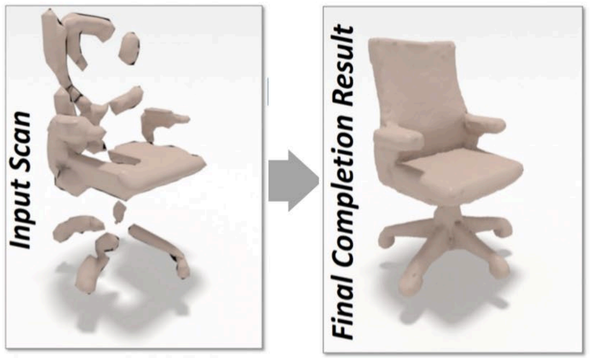


Image generation based on scene-graph

Single-view 3D reconstruction



Shape completion

Objective: learn a $p_{model}(y|x)$ that matches $p_{data}(y|x)$.

How to Learn Generative Models

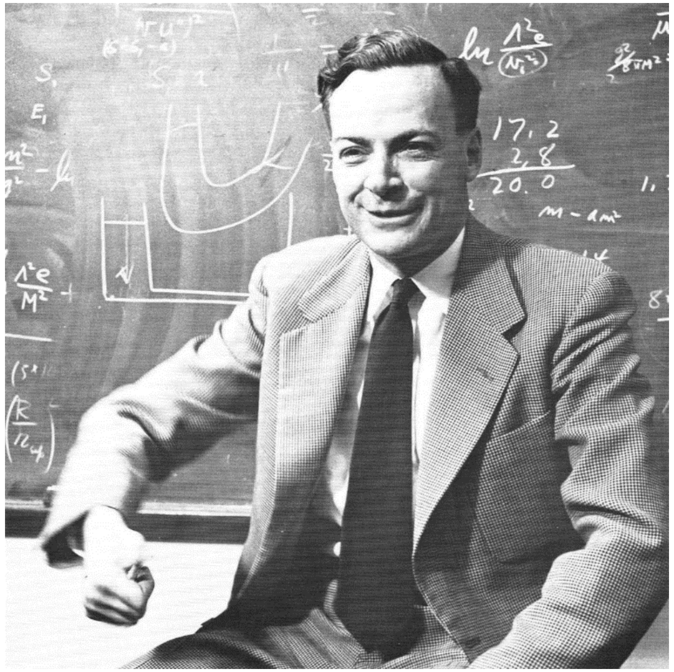
- Explicitly modeling data probabilistic density, learn a network $p_\theta(x)$ that maximize data probability

- Markov chain
- Autoregressive models
- **Variational autoencoder (VAE)**
- **Flow-based models**
- Energy based models
- …

- Implicitly modeling probabilistic density, e.g. learn a network that scores the realness of the data, $f_\theta(x)$

- **Generative adversarial network (GAN)**
- Score-based generative
- …

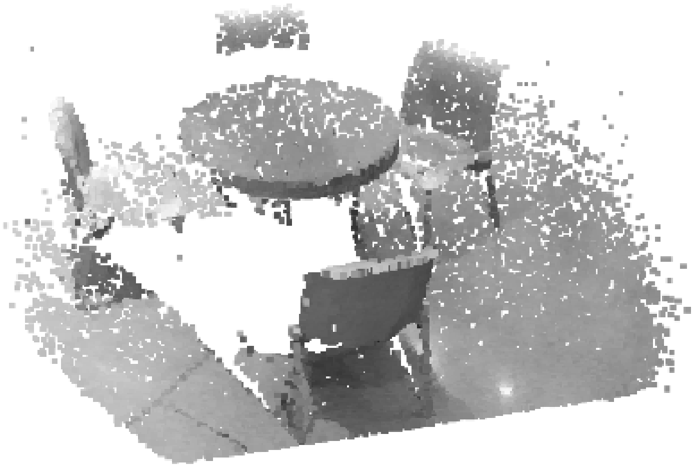
Generative Modeling



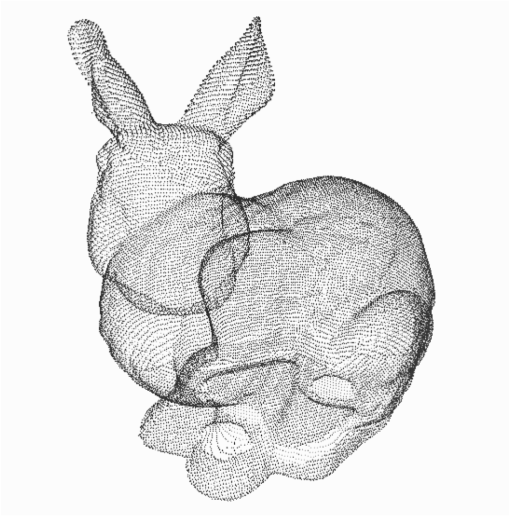
Richard Feynman: “*What I cannot create, I do not understand*”

Generative modeling: “*What I understand, I can create*”

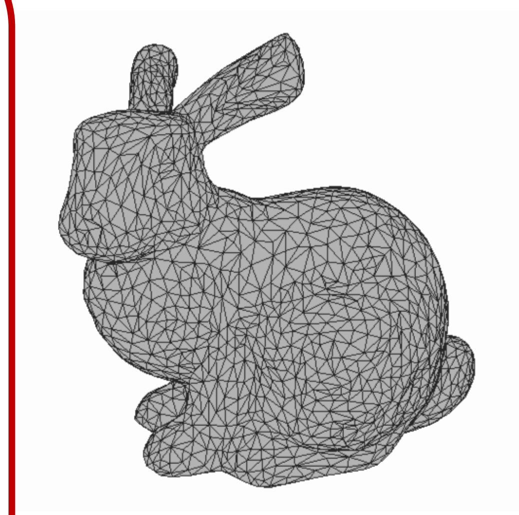
Background:
3D Representations
and Learning Frameworks



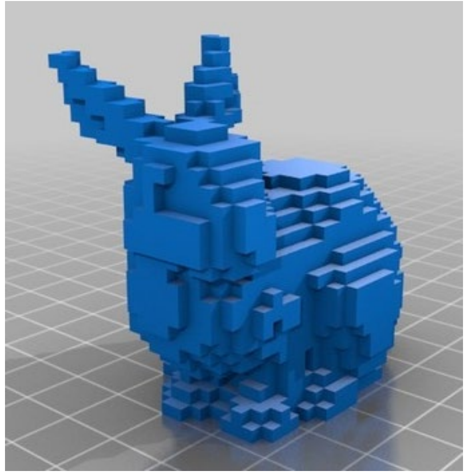
Multiple 3D Representations



Point Cloud



Surface Mesh



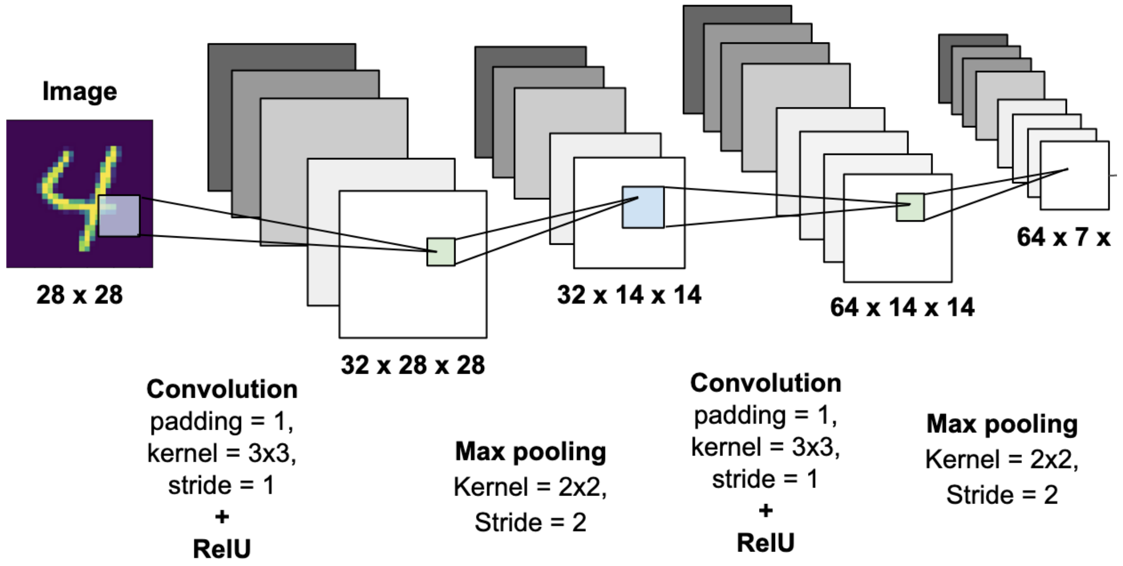
Volumetric

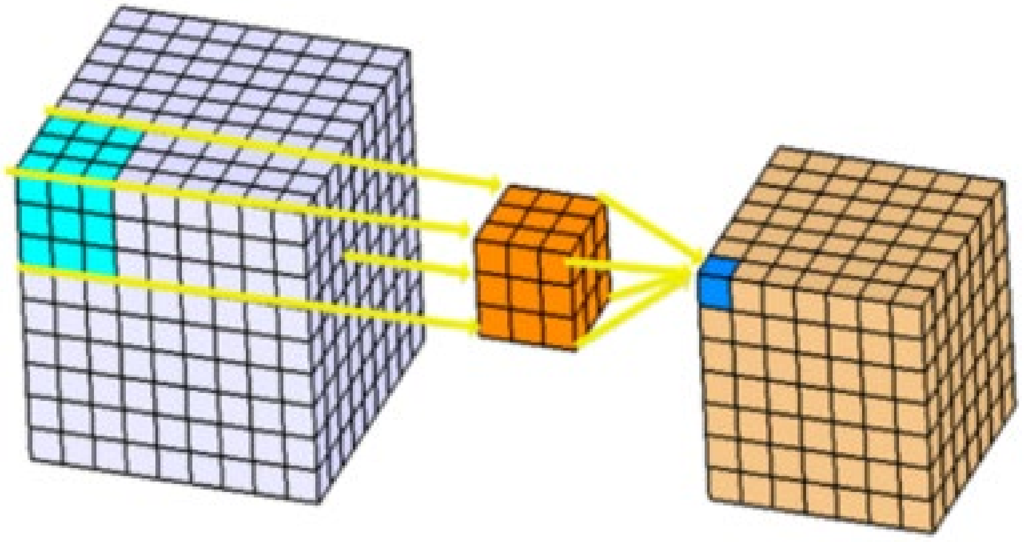
$$F(x) = 0$$

Implicit Shape

… (CSG, BSP, etc)

3D Convolution for Voxels





2D convolution

Kernel: K_h X K_w

Kernel weight: K_h X K_w X C_1 X C_2

Feature grid: H X W X C

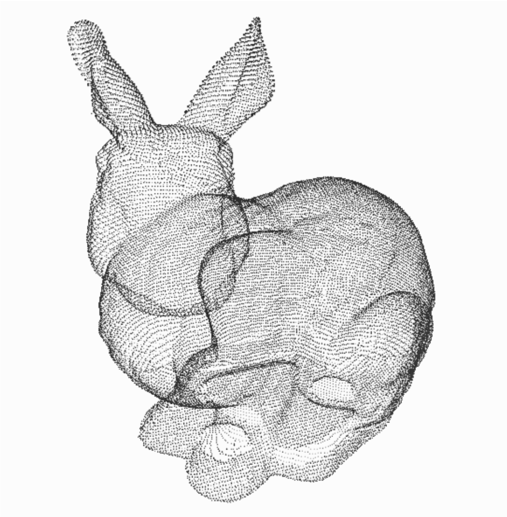
3D convolution

Kernel: K_h X K_w X K_d

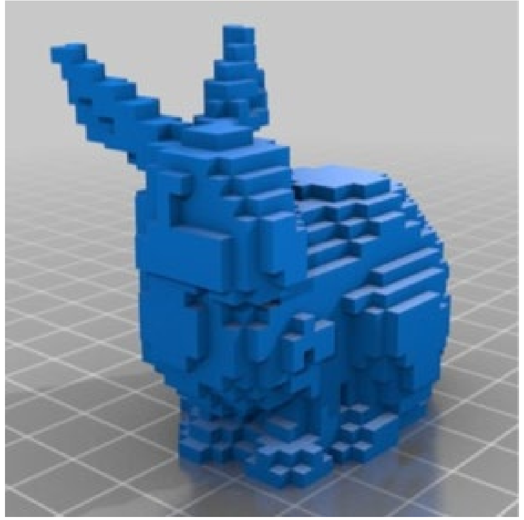
Kernel weight: K_h X K_w X K_d X C_1 X C_2

Feature grid: H X W X D X C

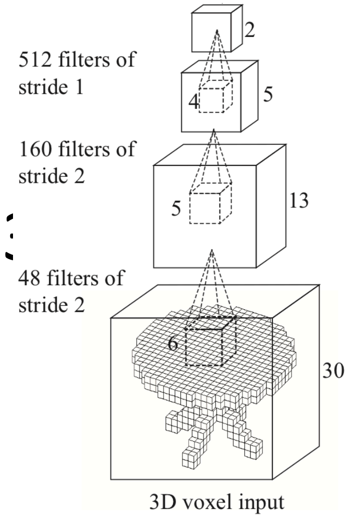
Summary: 3D Convolution



voxelization →



+

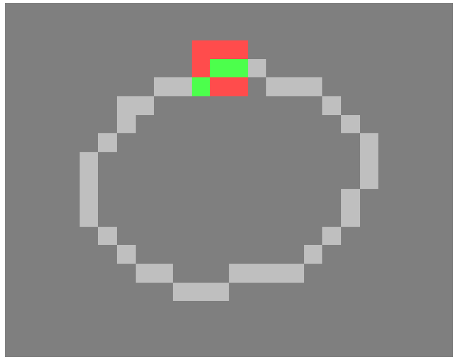


[Wu et al. 2015]

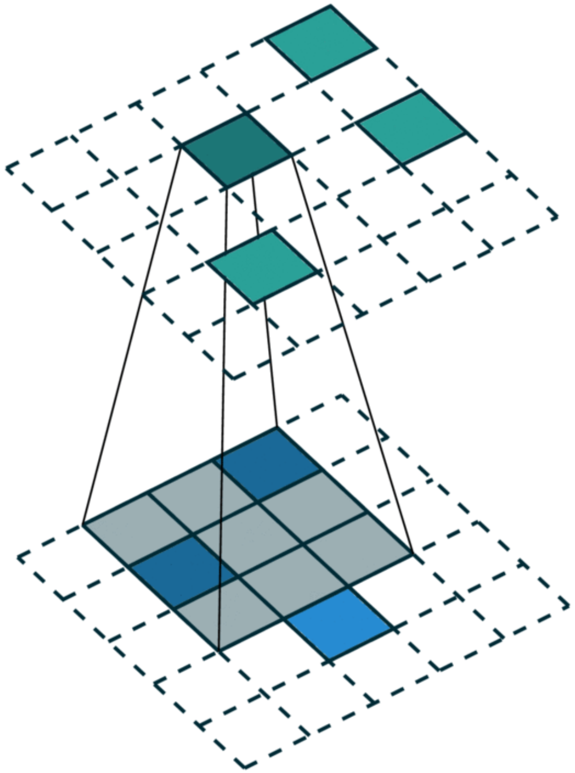
Con: High space complexity -- 3D convolution $O(N^3)$

Quantization errors in voxelization

Sparse Convolution



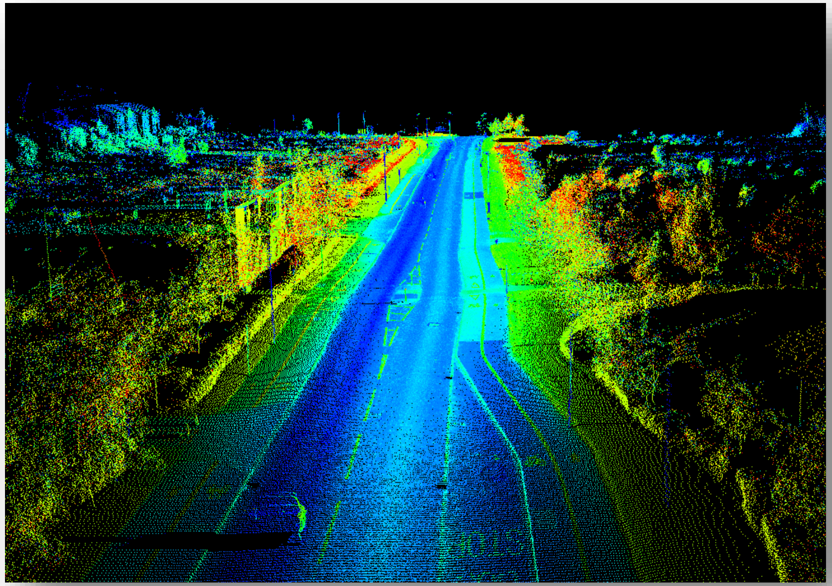
Submanifold sparse convolutional network (from FAIR)



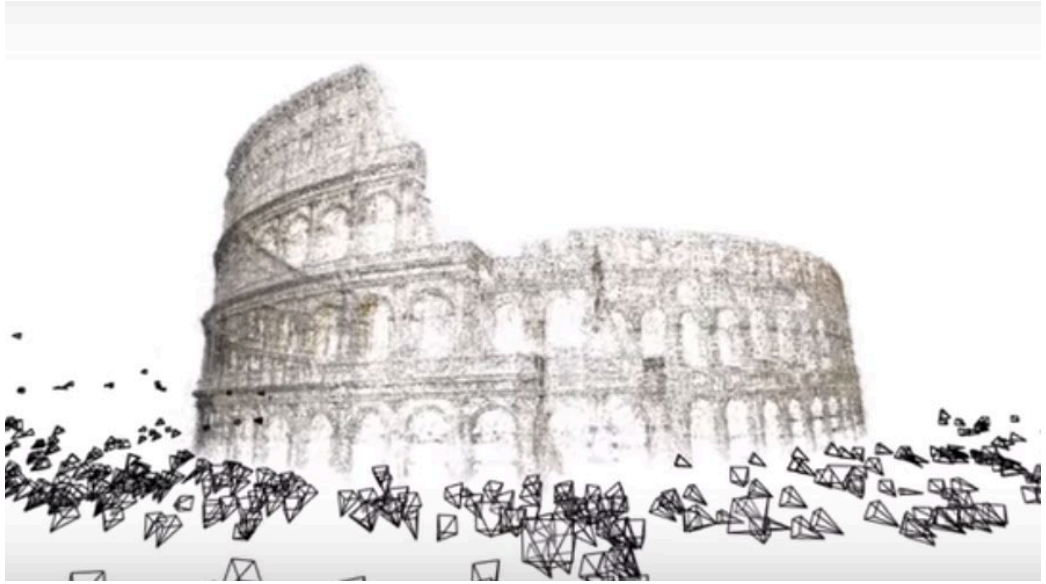
Minkowski Engine (from SVL)

Pro: computing efficiently. Con: still quantization.

Point Clouds from Many Sensors



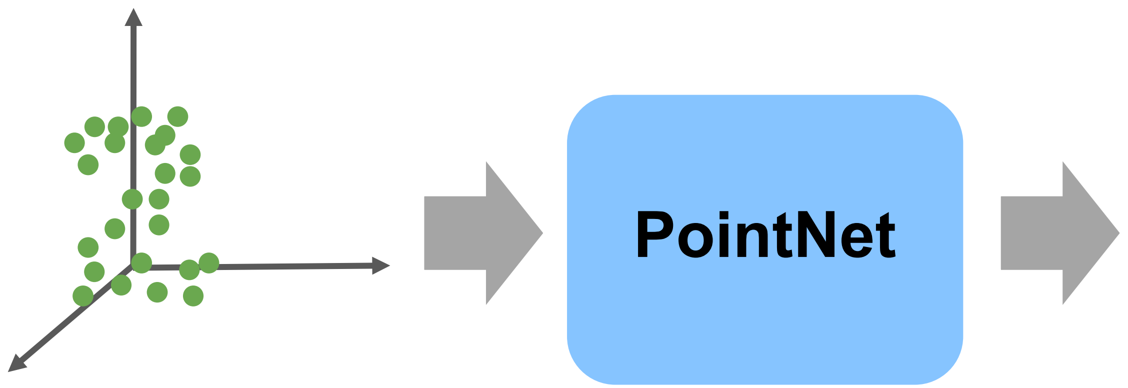
Lidar point clouds (LizardTech)



Structure from motion (Microsoft)

Depth camera (Intel)

PointNet: First Learning Tool for Point Clouds



Object Classification

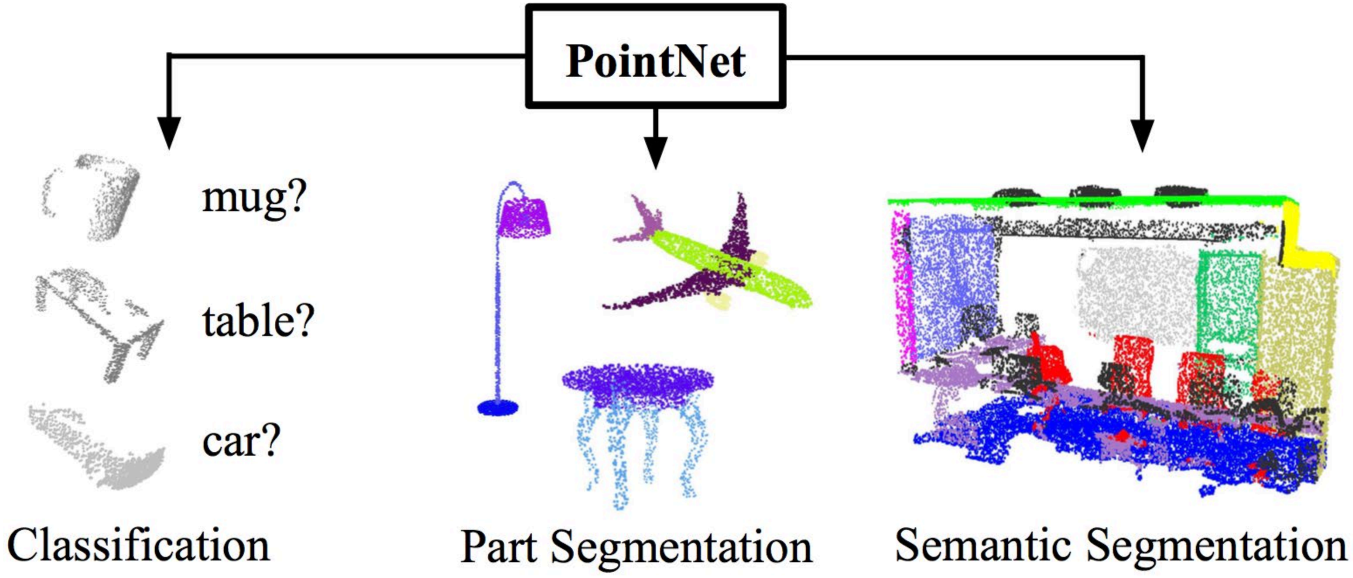
Object Part Segmentation

Semantic Scene Parsing

...

End-to-end learning for irregular point data

Unified framework for various tasks



Charles R. Qi, Hao Su, Kaichun Mo, Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. (CVPR'17)

Invariances

The model has to respect key desiderata for point clouds:

Point Permutation Invariance

Point cloud is a set of unordered points

Sampling Invariance

Output a function of the underlying geometry and not the sampling

Permutation Invariance: Symmetric Functions

$$f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D$$

Examples:

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

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

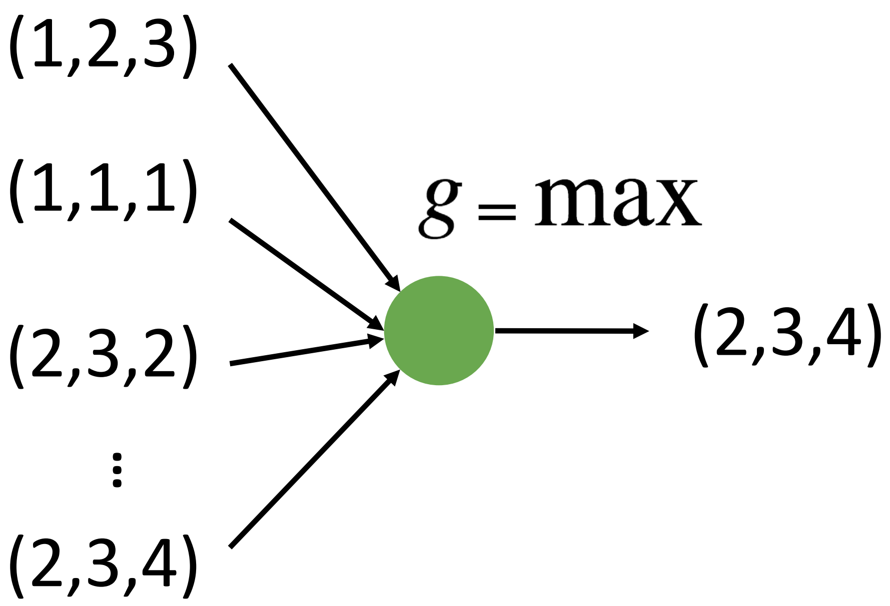
$\ldots$

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

Construct Symmetric Functions by Neural Networks

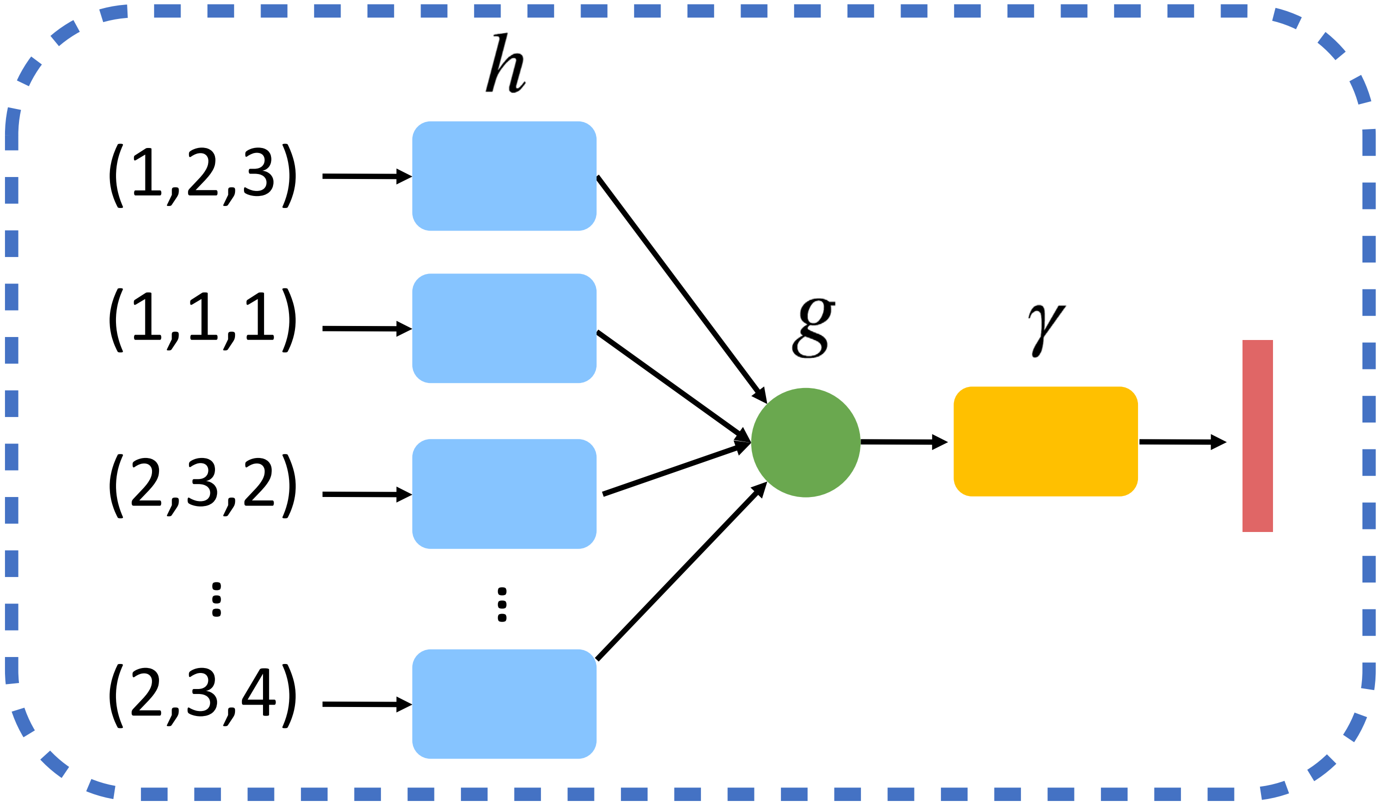
Simplest form: directly aggregate all points with a symmetric operator g

Just discovers simple extreme/aggregate properties of the geometry.



Construct Symmetric Functions by Neural Networks

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



PointNet (vanilla)

Distance Metrics for Point Cloud

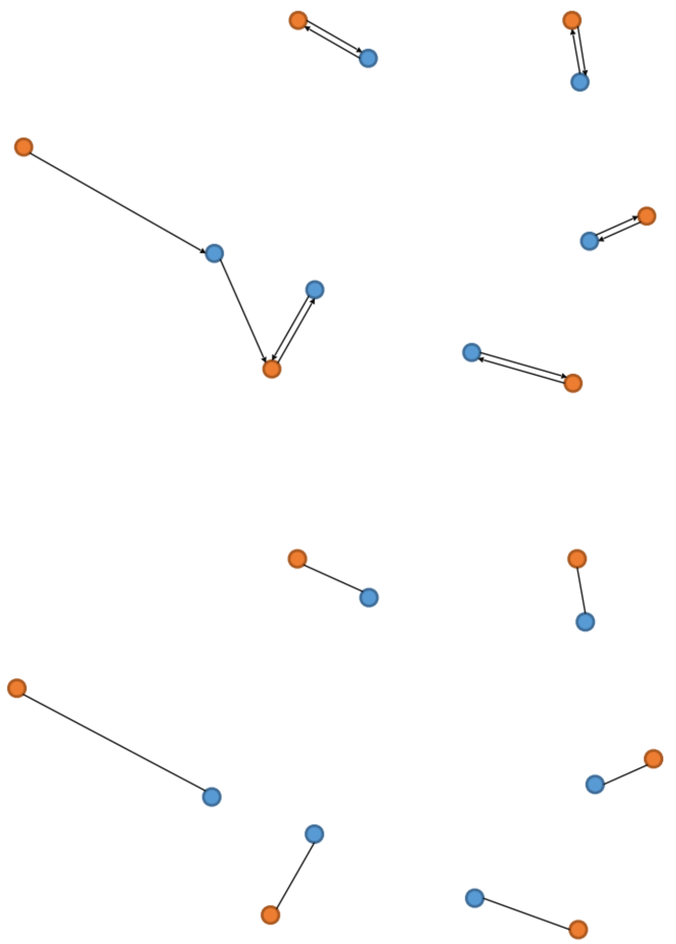
Chamfer distance We define the Chamfer distance between $S_1, S_2 \subseteq \mathbb{R}^3$ as:

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

Earth Mover's distance Consider $S_1, S_2 \subseteq \mathbb{R}^3$ of equal size $s = |S_1| = |S_2|$. The EMD between A and B is defined as:

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

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



A Point Set Generation Network for 3D Object Reconstruction from a Single Image, CVPR 2016

Distance Metrics for Point Cloud

Chamfer distance We define the Chamfer distance between $S_1, S_2 \subseteq \mathbb{R}^3$ as:

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

Sum of closest distances

Insensitive to sampling

Earth Mover's distance Consider $S_1, S_2 \subseteq \mathbb{R}^3$ of equal size $s = |S_1| = |S_2|$. The EMD between A and B is defined as:

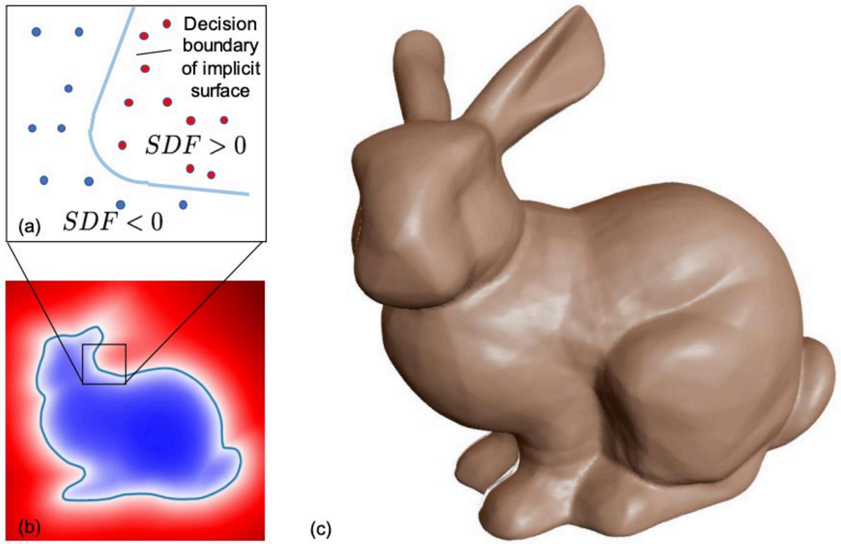
$$d_{EMD}(S_1, S_2) = \min_{\phi : S_1 \to S_2} \sum_{x \in S_1} \|x - \phi(x)\|_2$$

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

Sum of matched closest distances

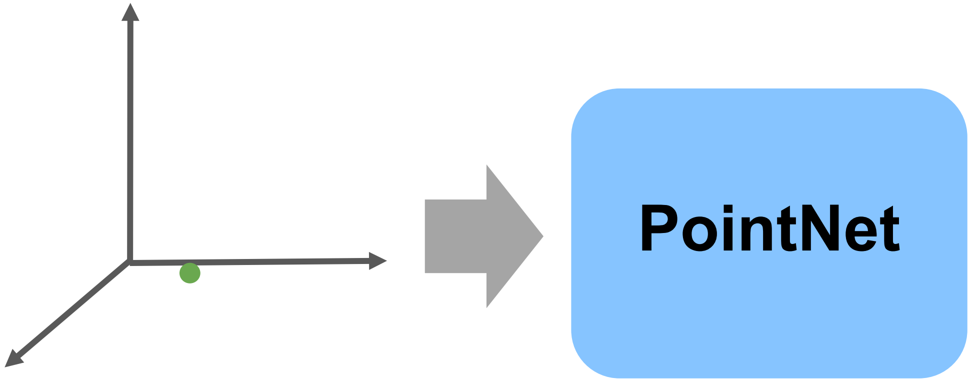
Sensitive to sampling

Convolution on Implicit Functions

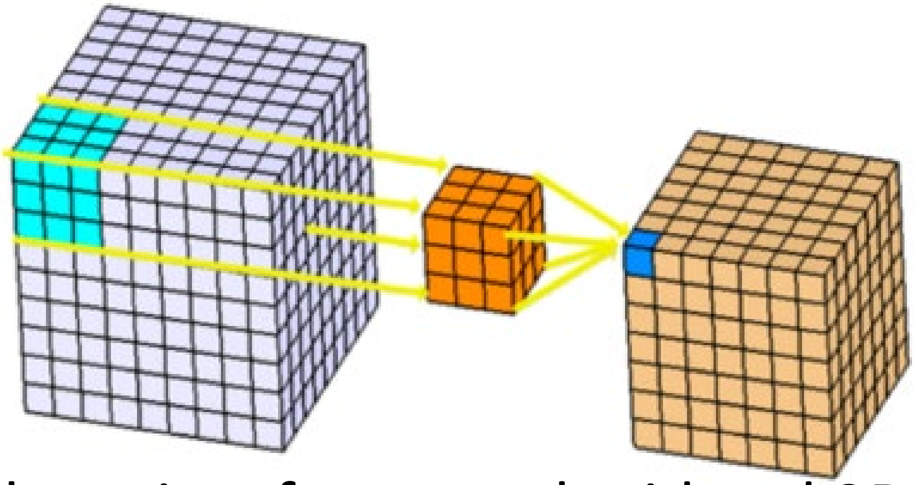


SDF is a scalar field.

Convolution on implicit function is still very immature.

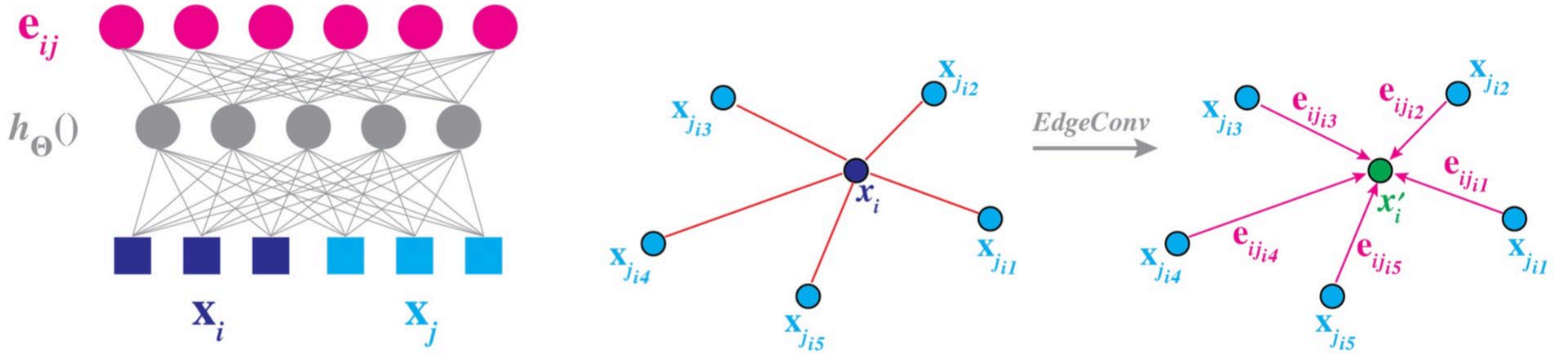


Sample points and use PointNet to extract features.



Take points from voxel grid and 3D convolution to extract features.

Convolution on Mesh/Graph

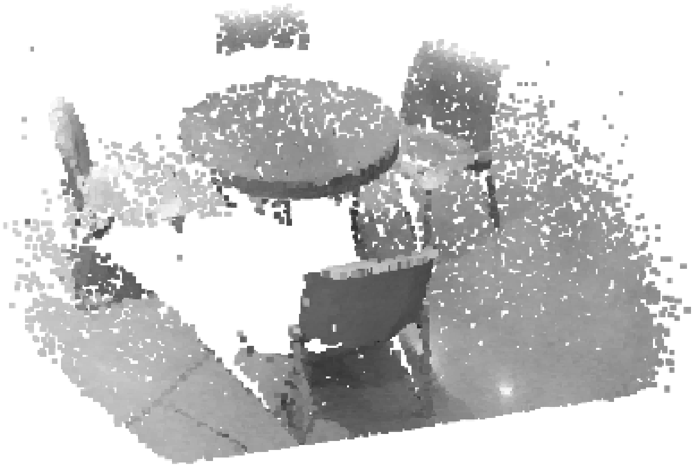


Message passing: The output of EdgeConv at the i-th vertex is thus given by

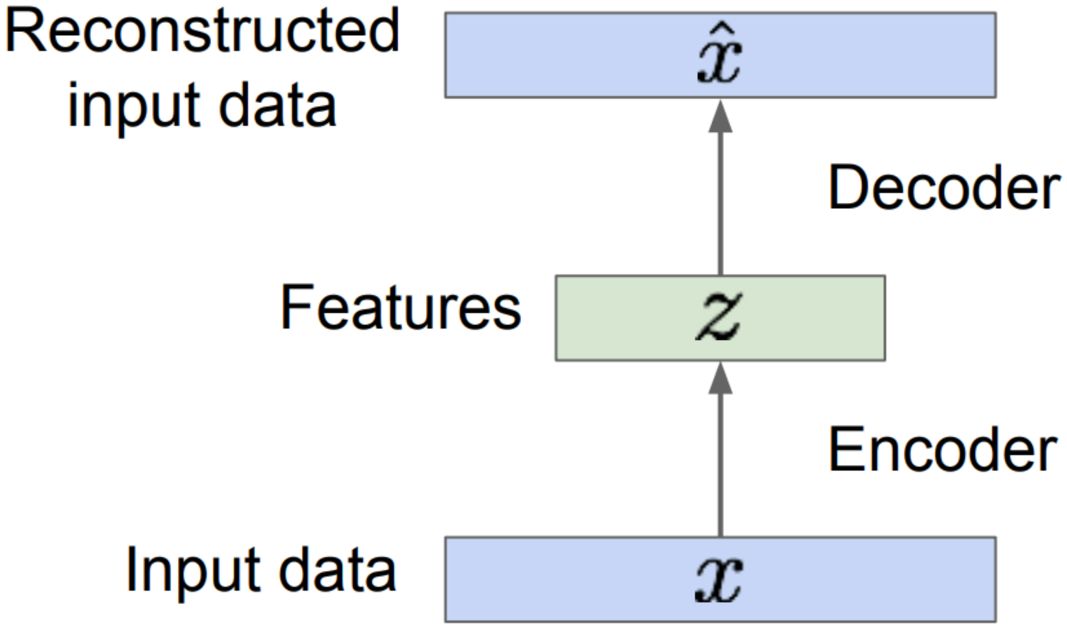
$$\mathbf{x}'_i = \mathop{\square}_{j:(i,j)\in\mathcal{E}} h_\Theta(\mathbf{x}_i, \mathbf{x}_j). \qquad (1)$$

Wang, et.al., Dynamic Graph CNN for Learning on Point Clouds, ToG 2019

Deep 3D Generative Models



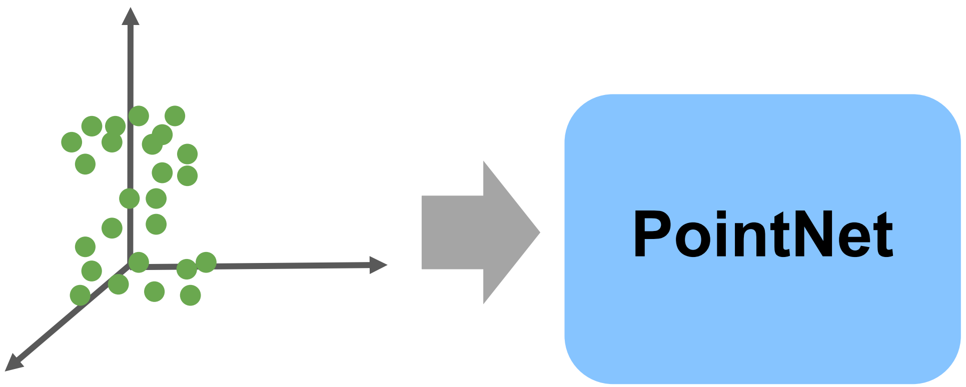
Auto-Encoder

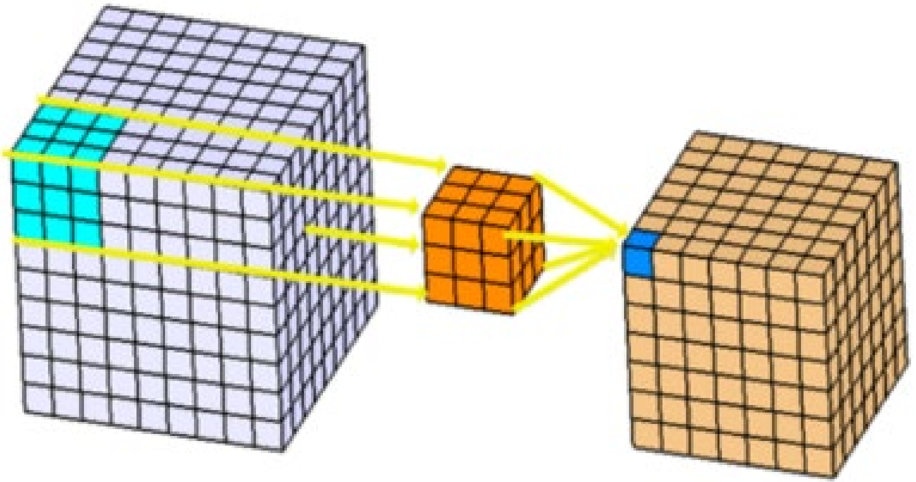


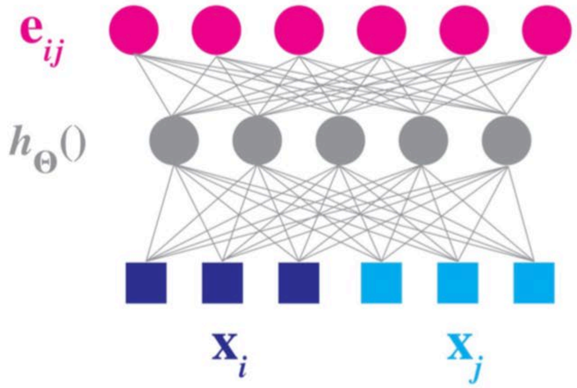
- AE encodes itself into a latent z
- AE then decodes the latent z back to itself
- Understanding AE is the first step to understand generative models.

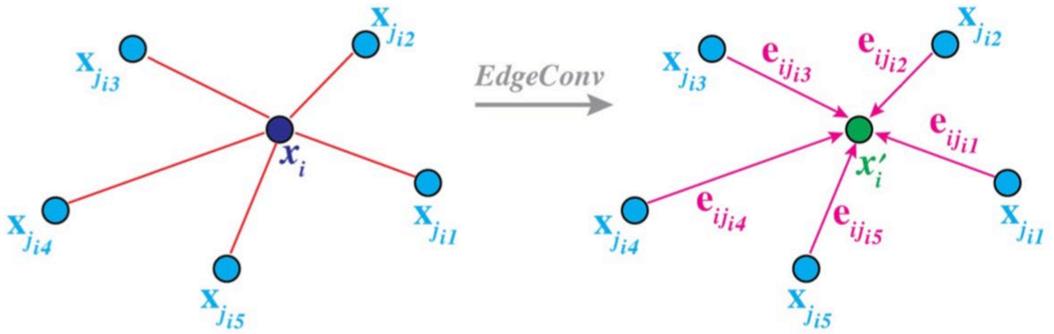
Image Credit: Stanford CS231N

Encoding 3D using Convolution



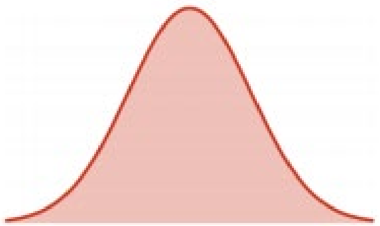


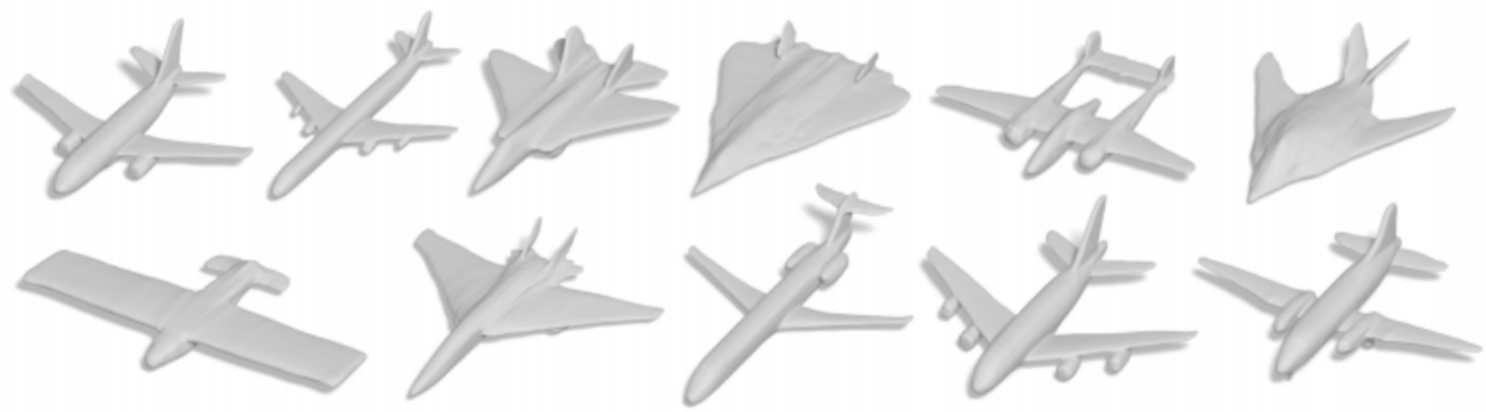




Encoding: Convolution networks can transform a 3D data into a vector in latent space.

Decoding/Generation



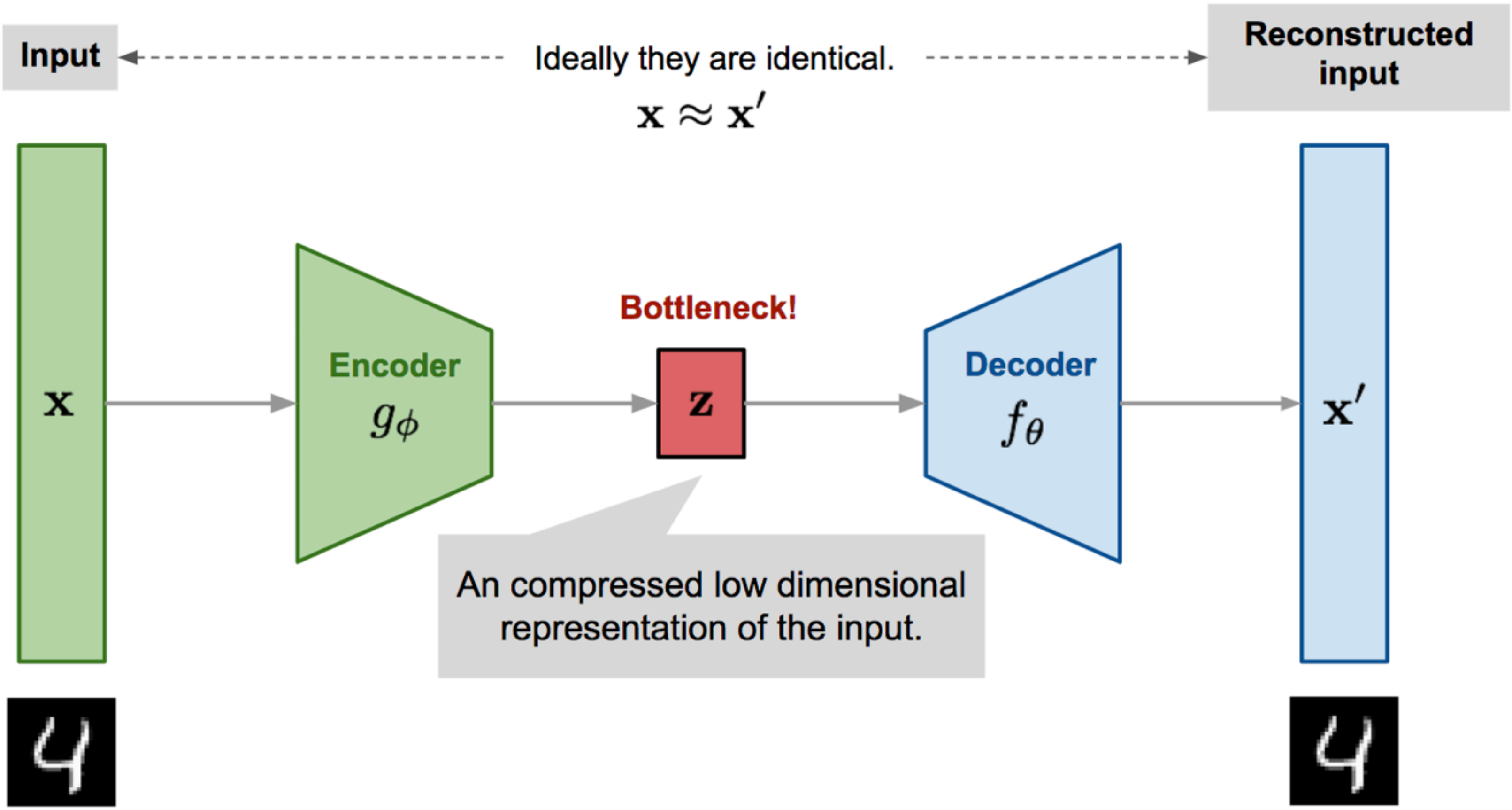


Latent vectors *z*

Generated Shapes

Generator/Decoder: generating shapes from latent vectors

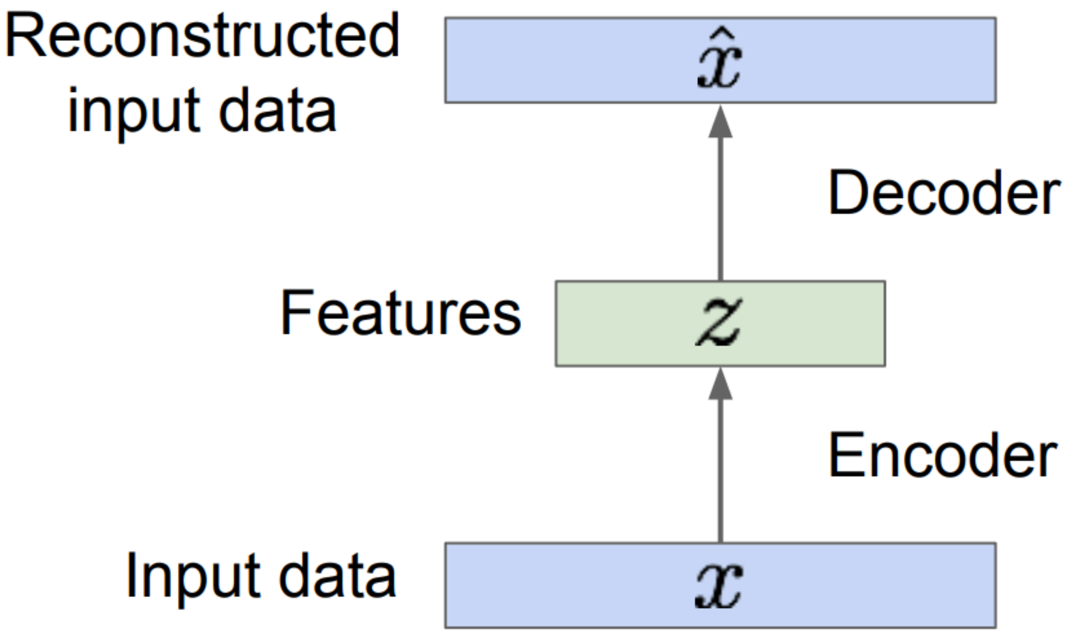
Auto-Encoder



Task: Learn to encode the input and decode itself

Reconstruction loss: measuring the distance between the input/output

Auto-Encoder

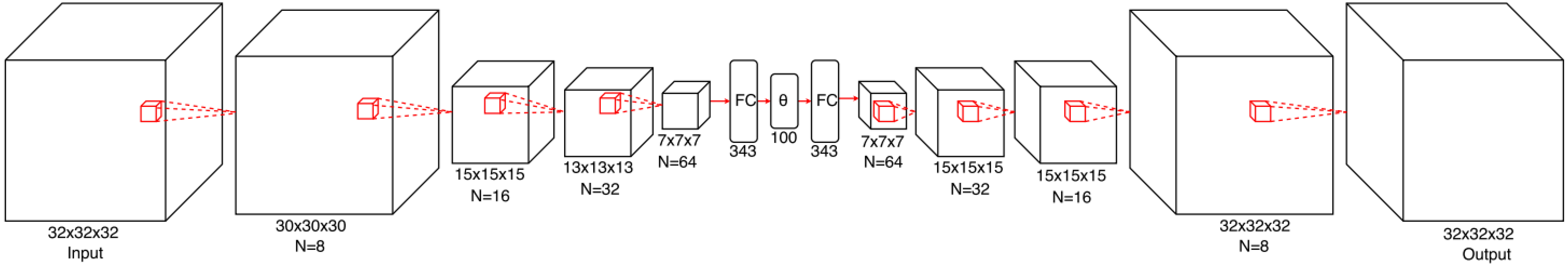


- Why dimension reduction?
  Want features to capture meaningful factors of variation in data.

- Unsupervised/Self-supervised

Image Credit: Stanford CS231N

Volumetric AE

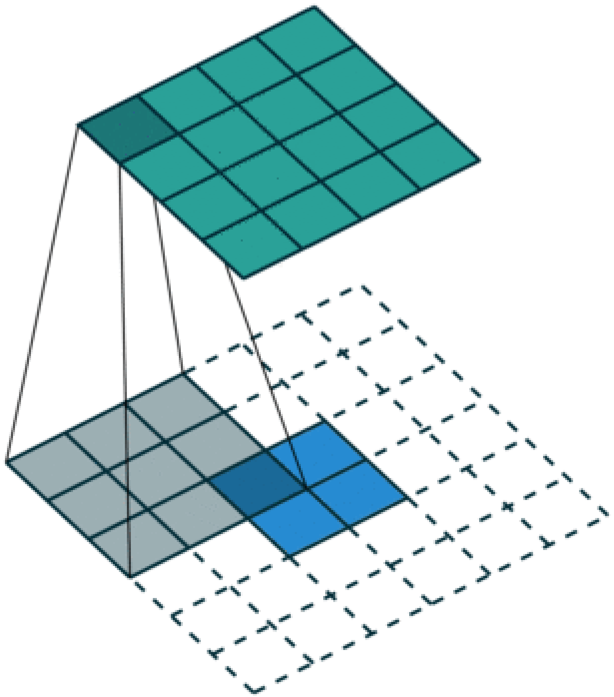


Binary Cross-Entropy Loss: $\mathcal{L} = -t\,log(o) \; - \; (1-t)\,log(1-o)$

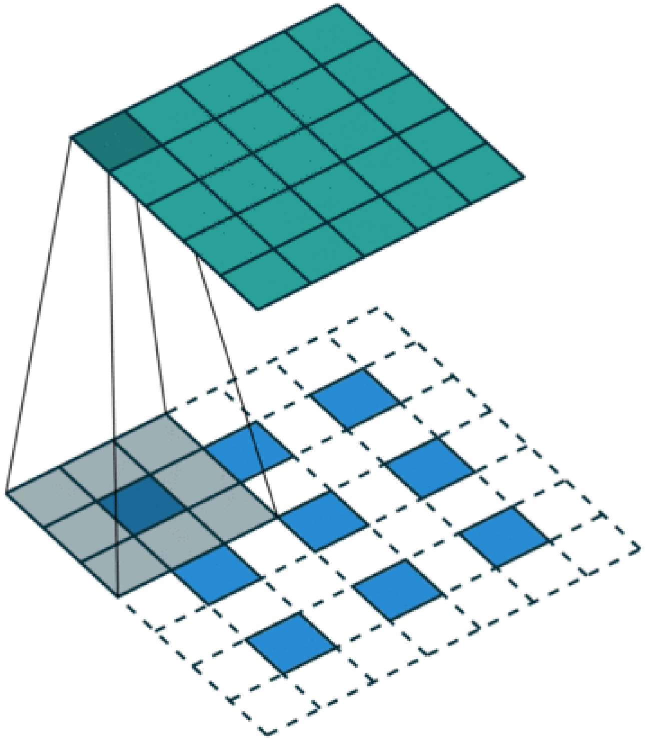
CoRR 2016

Generative and Discriminative Voxel Modeling with Convolutional Neural Networks

Deconvolution (Transposed Conv)



Stride = 1, Padding = 0



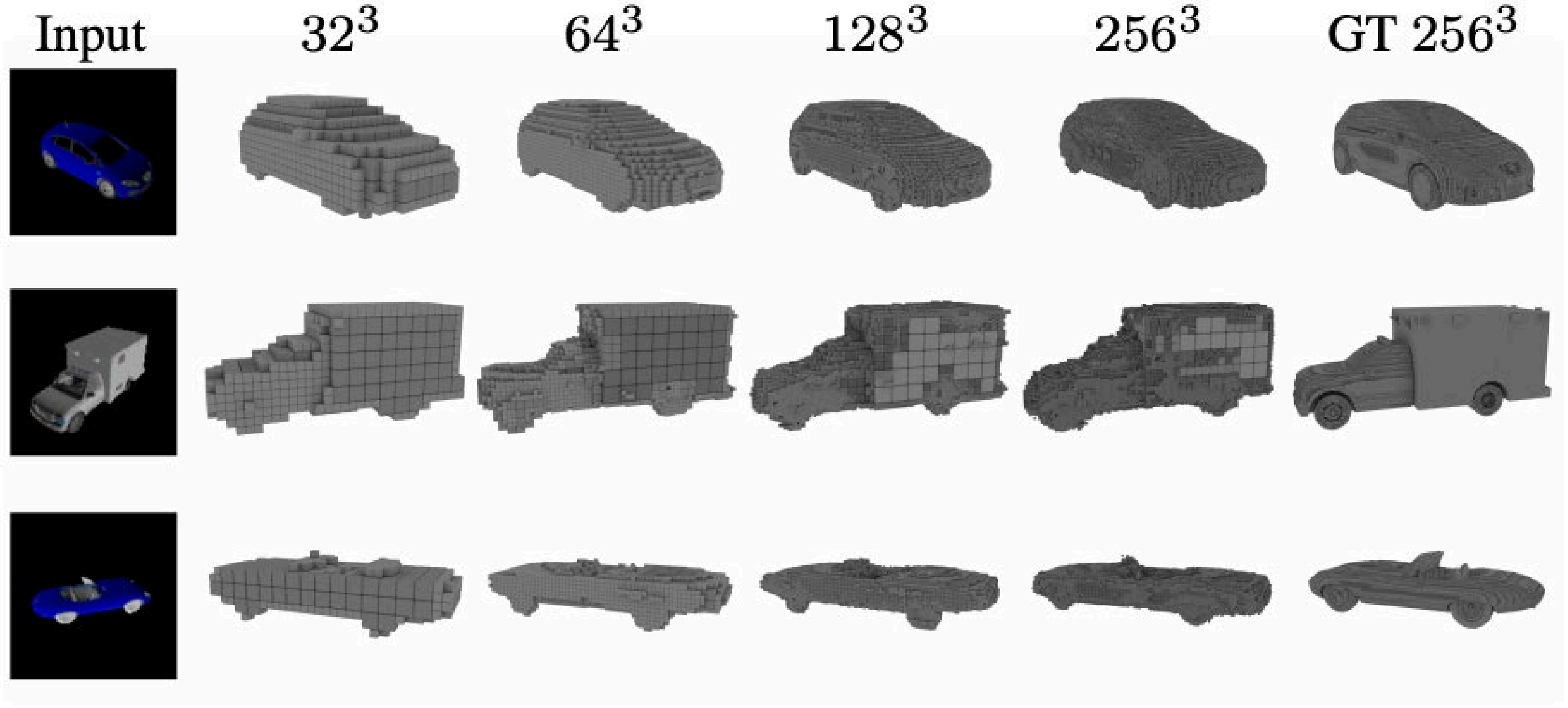
Stride = 2, Padding = 1

Image credit:
https://github.com/vdumoulin/conv_arithmetic

Auto-Encoder Connecting 2D and 3D

Encoder: 2D Conv

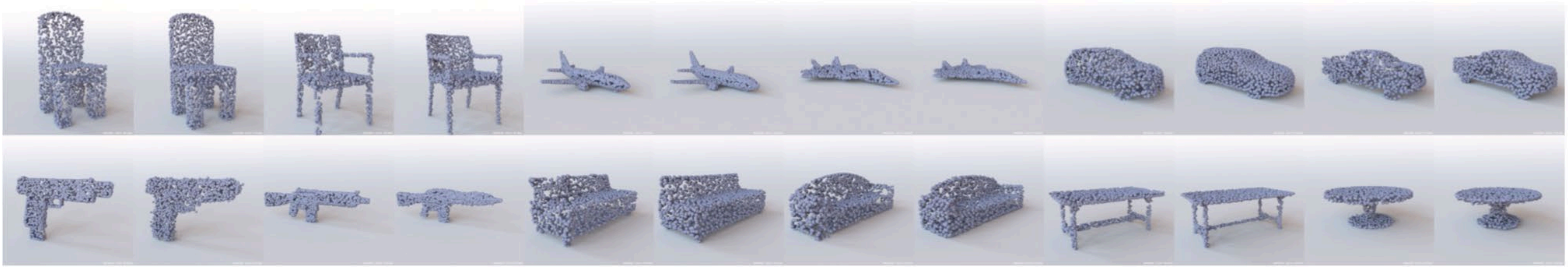
Decoder: 3D Deconv (Octree decoder)



Tatarchenko et al., "**Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs**", *ICCV 2017*

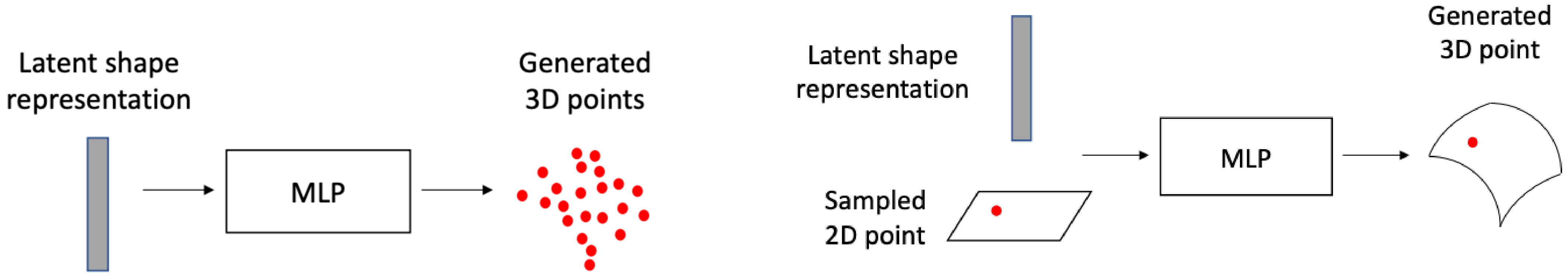
Point Cloud AE

Encoder: PointNet (N*3 → L)
Decoder: MLP (L → 3N → N*3)



ICML, 2018, Learning Representations and Generative Models for 3D Point Clouds, Panos Achlioptas, et. al.

Parametric Decoder: AtlasNet

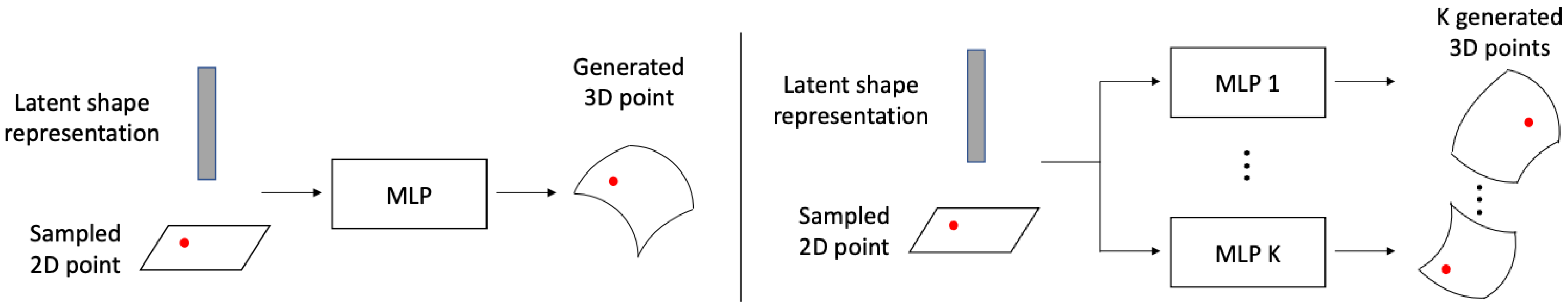


Given that the output points form a smooth surface, enforce such a parametrization in input. For each point (u, v) on the parameterization, MLP([z, uv]) -> point

Also, you can get **Mesh!**

AtlasNet: A Papier-Machˆe Approach to Learning 3D Surface Generation, CVPR 2018

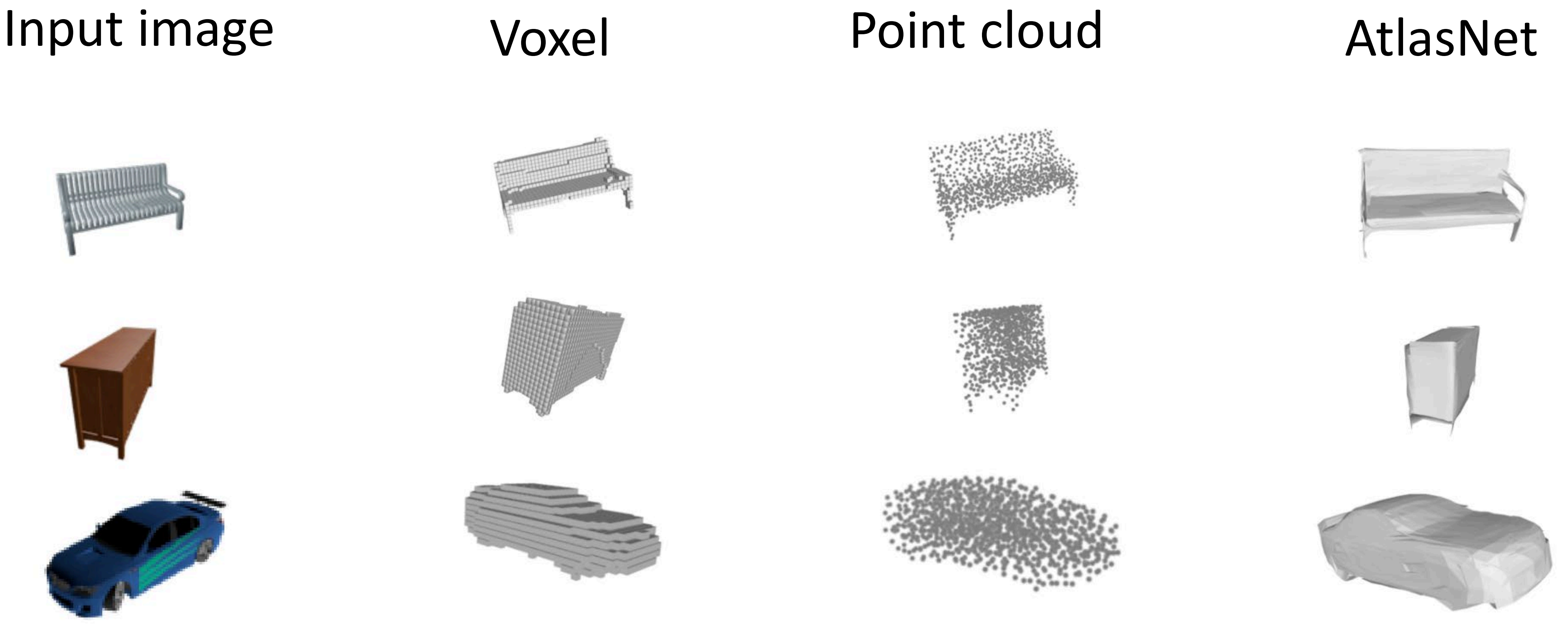
Parametric Decoder: AtlasNet



One parameterization (an **atlas**) is limited for objects with complex topology.
So, **more sheets.**

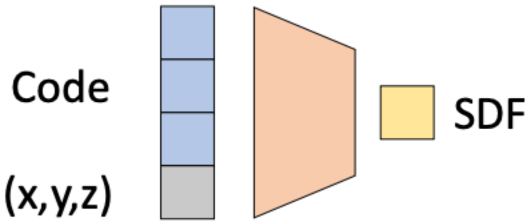
AtlasNet: A Papier-Machˆe Approach to Learning 3D Surface Generation, CVPR 2018

Comparison



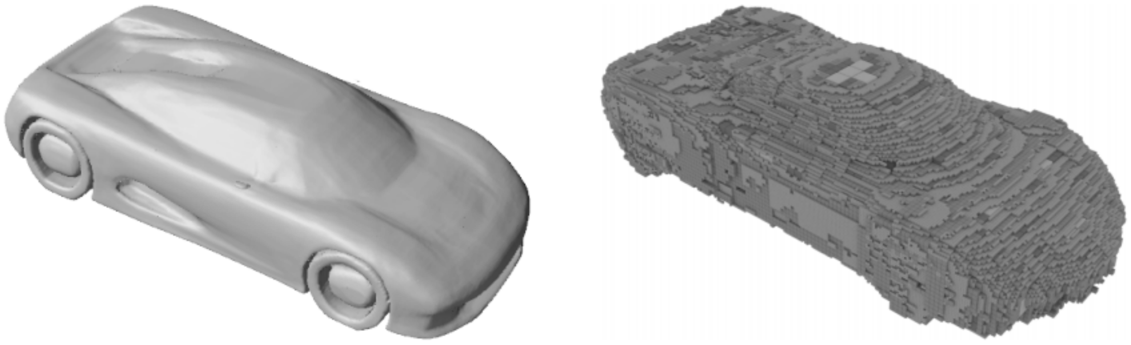
AtlasNet: A Papier-Machˆe Approach to Learning 3D Surface Generation, CVPR 2018

AutoEncoding SDF: Deep SDF



Decoder

Comparison with Octree

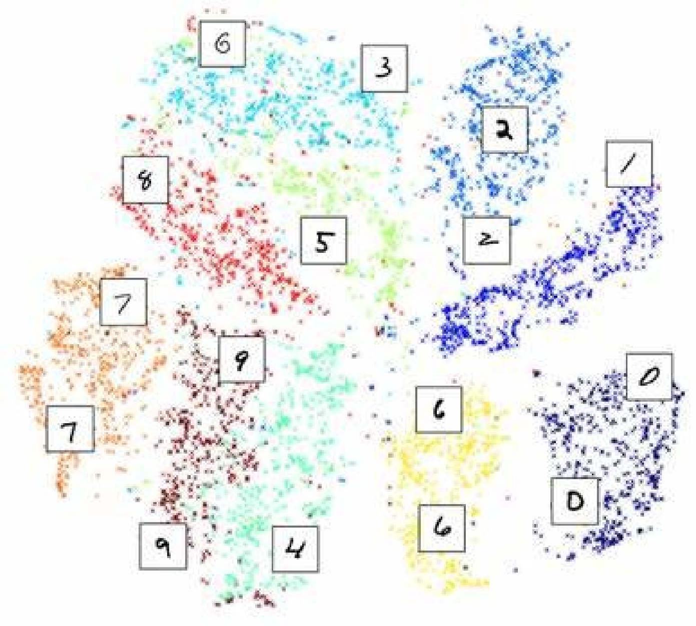


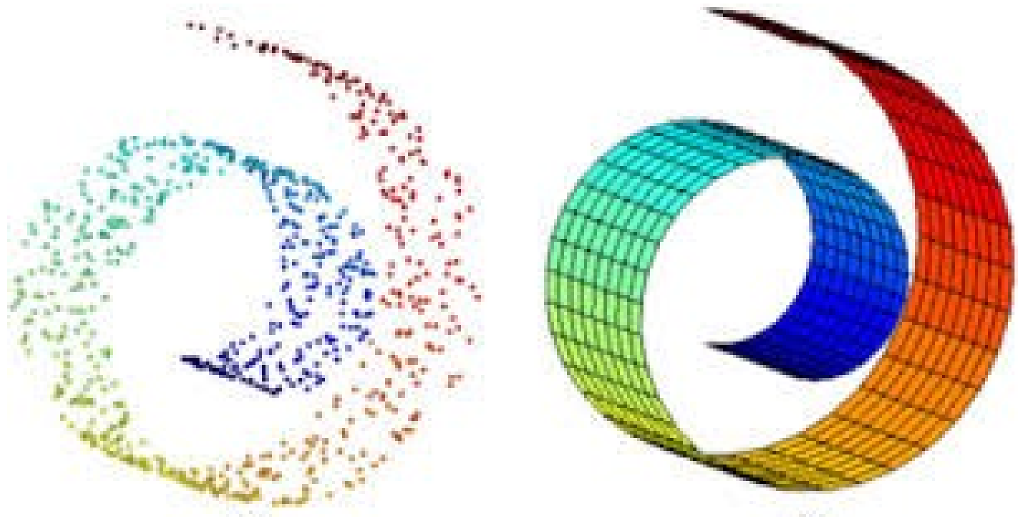
(a) Ground-truth **(b)** Our Result **(c)** [22]-25 patch **(d)** [22]-sphere

DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, CVPR 2019

Discussion

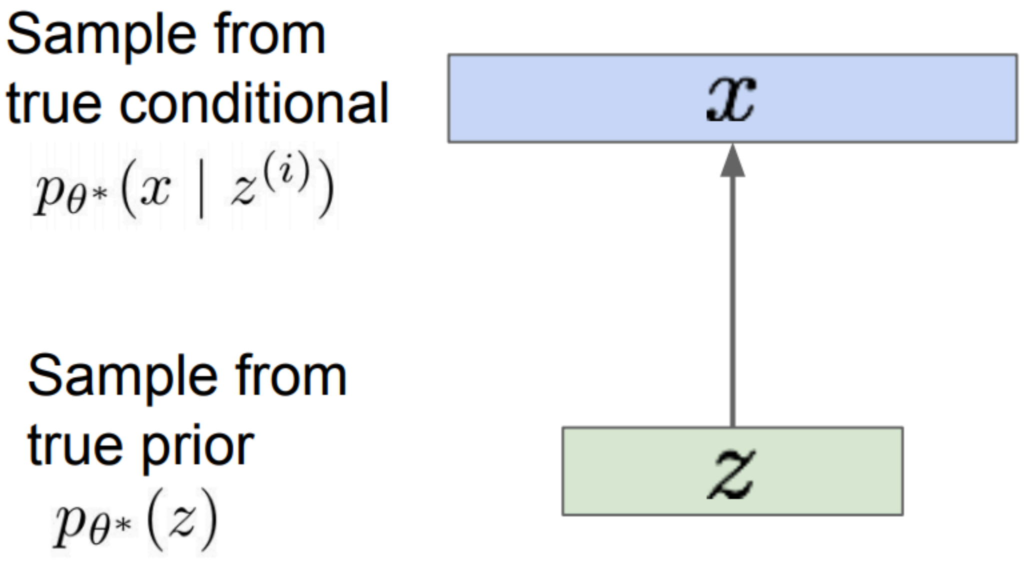
- Where is the data manifold in the latent space?
- Is a vanilla autoencoder a generative model?





(a) (b)

Variational Auto-Encoder

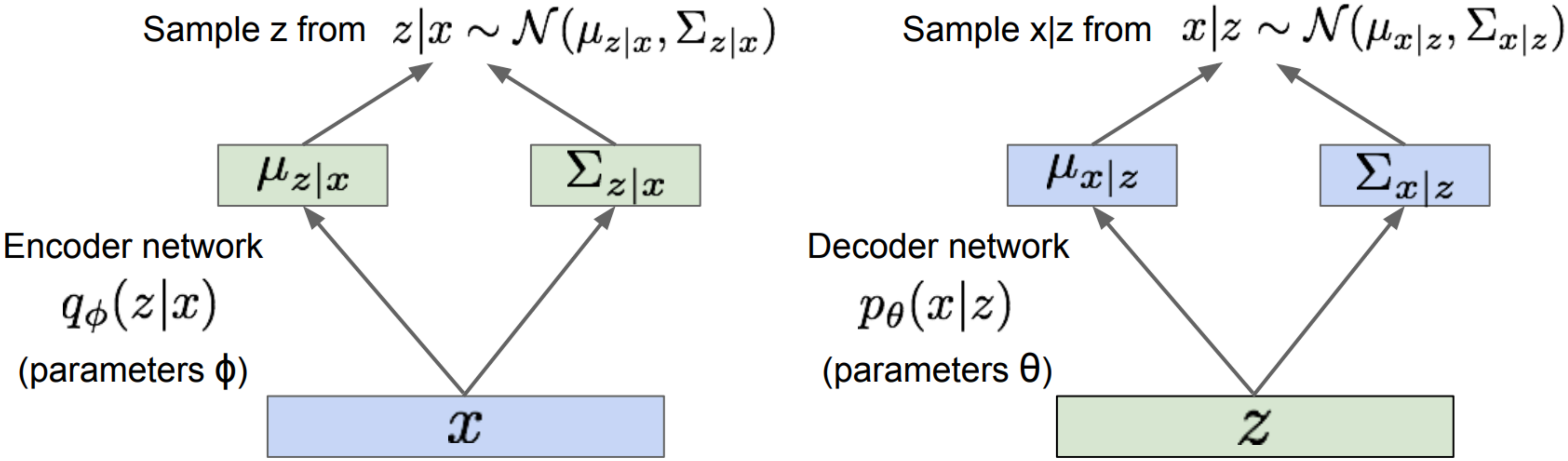


We can assume z follows a distribution.

Choose prior p(z) to be simple, e.g. Gaussian.

Image Credit: Stanford CS231N

Variational Auto-Encoder



Encoder

Decoder

Image Credit: Stanford CS231N

Training VAE

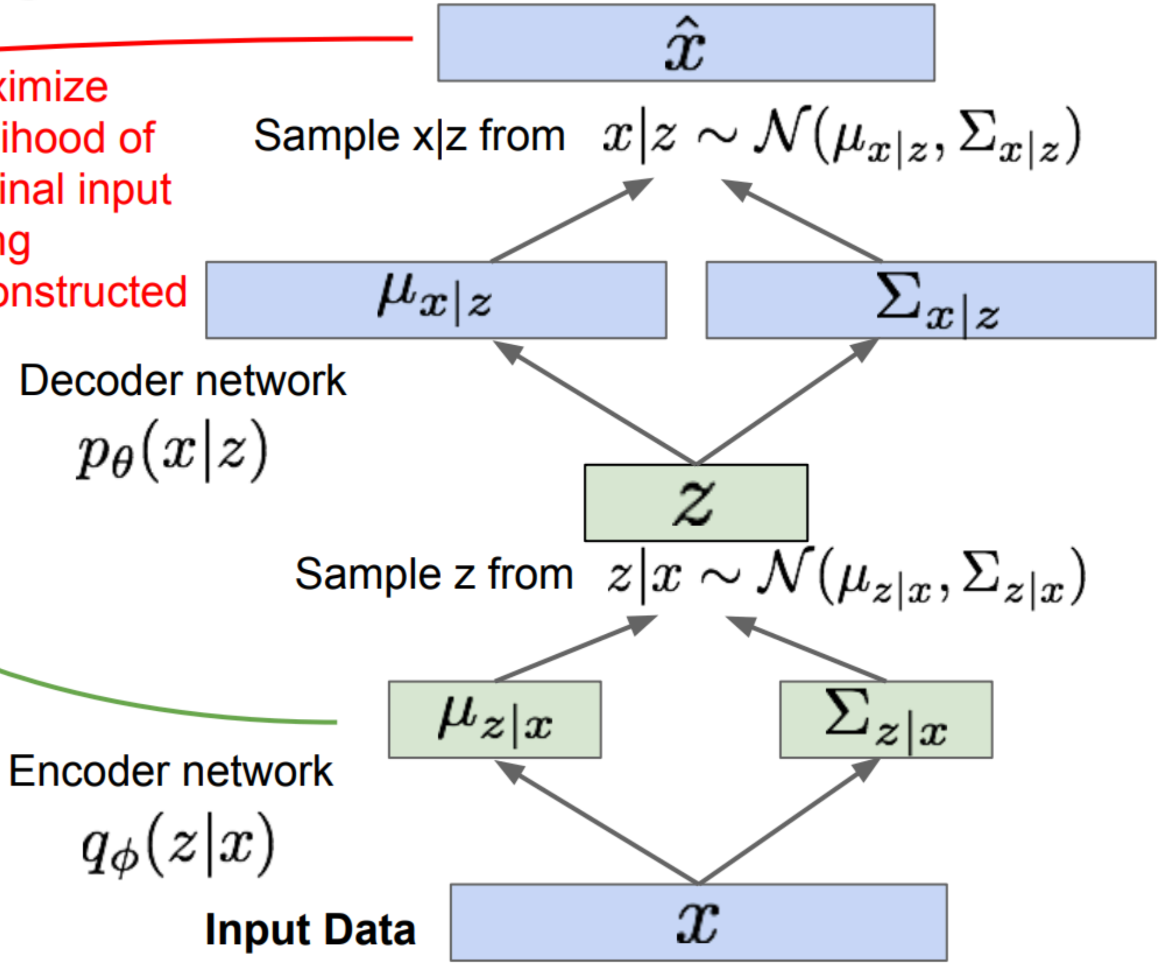
Variational Autoencoders

Putting it all together: maximizing the likelihood lower bound

$$\underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)}$$

Maximize likelihood of original input being reconstructed

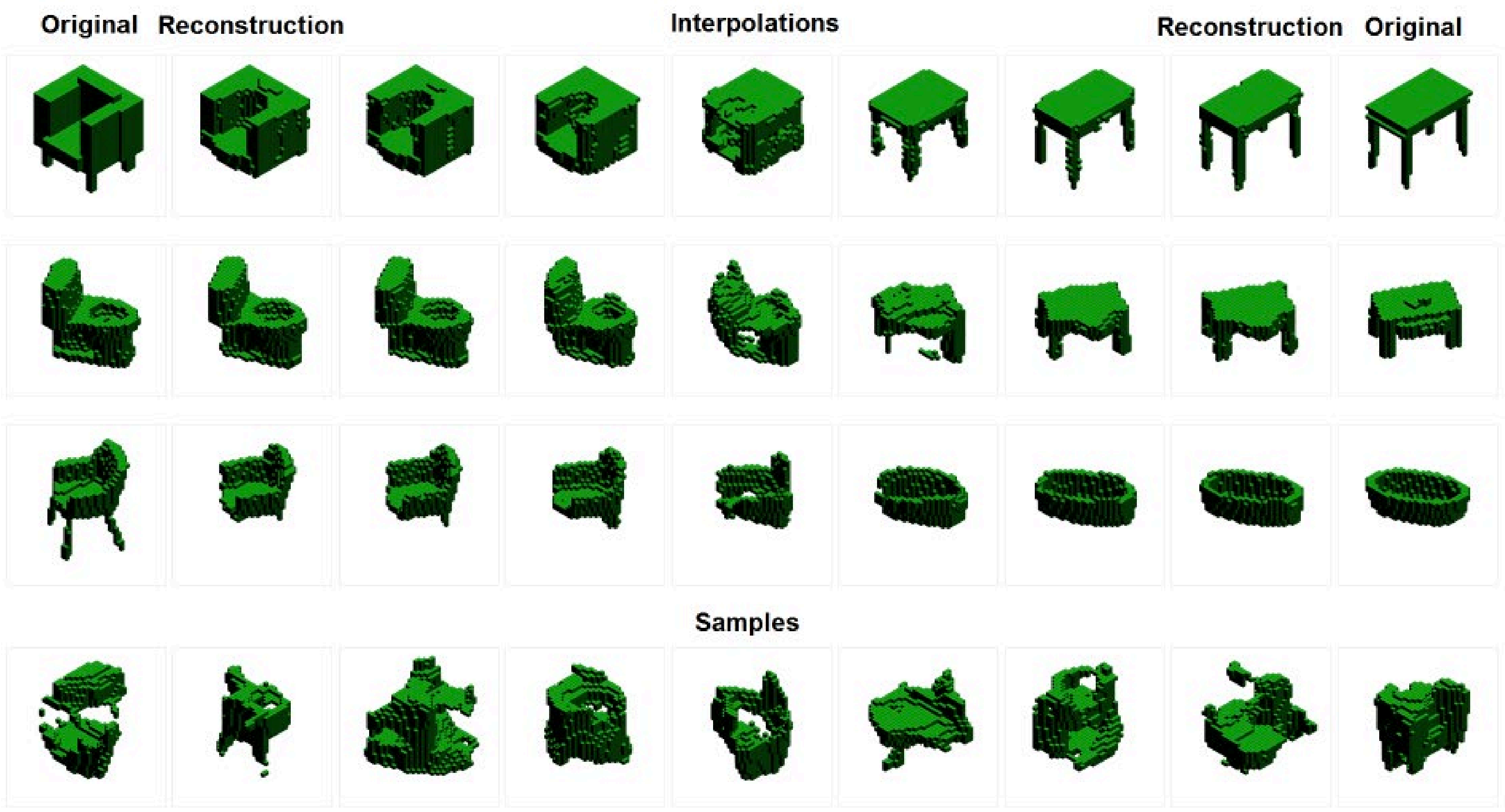
Make approximate posterior distribution close to prior



For every minibatch of input data: compute this forward pass, and then backprop!

Credit: Stanford CS231N

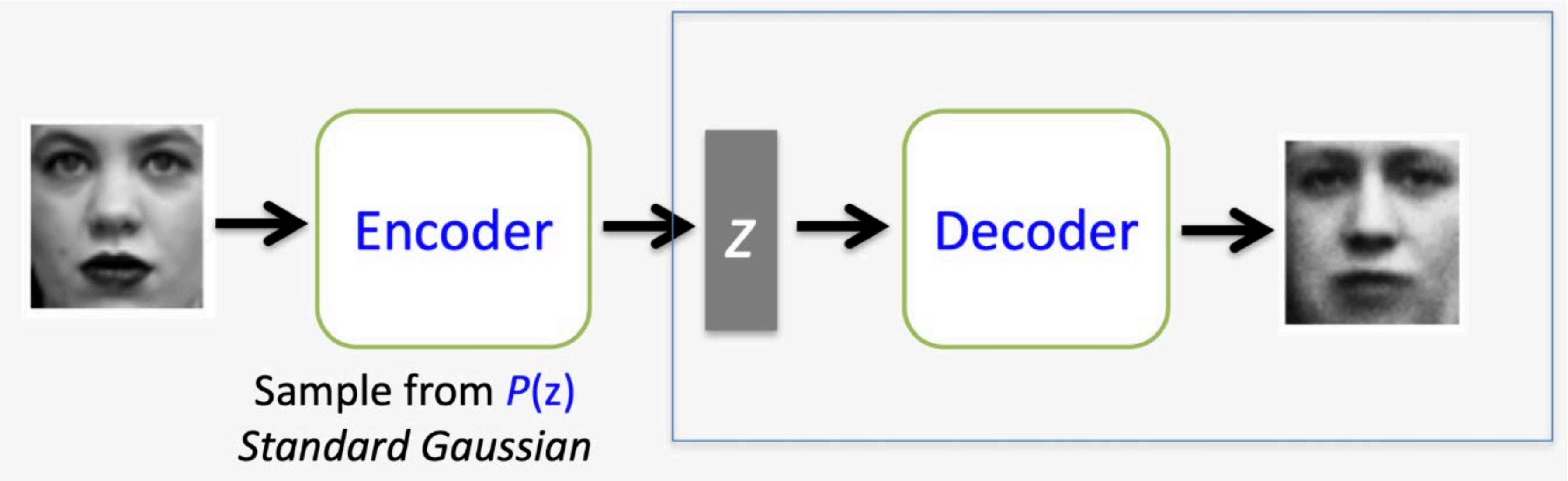
Generating New Samples & Interpolation



CoRR 2016

Generative and Discriminative Voxel Modeling with Convolutional Neural Networks

Issues for Autoencoders

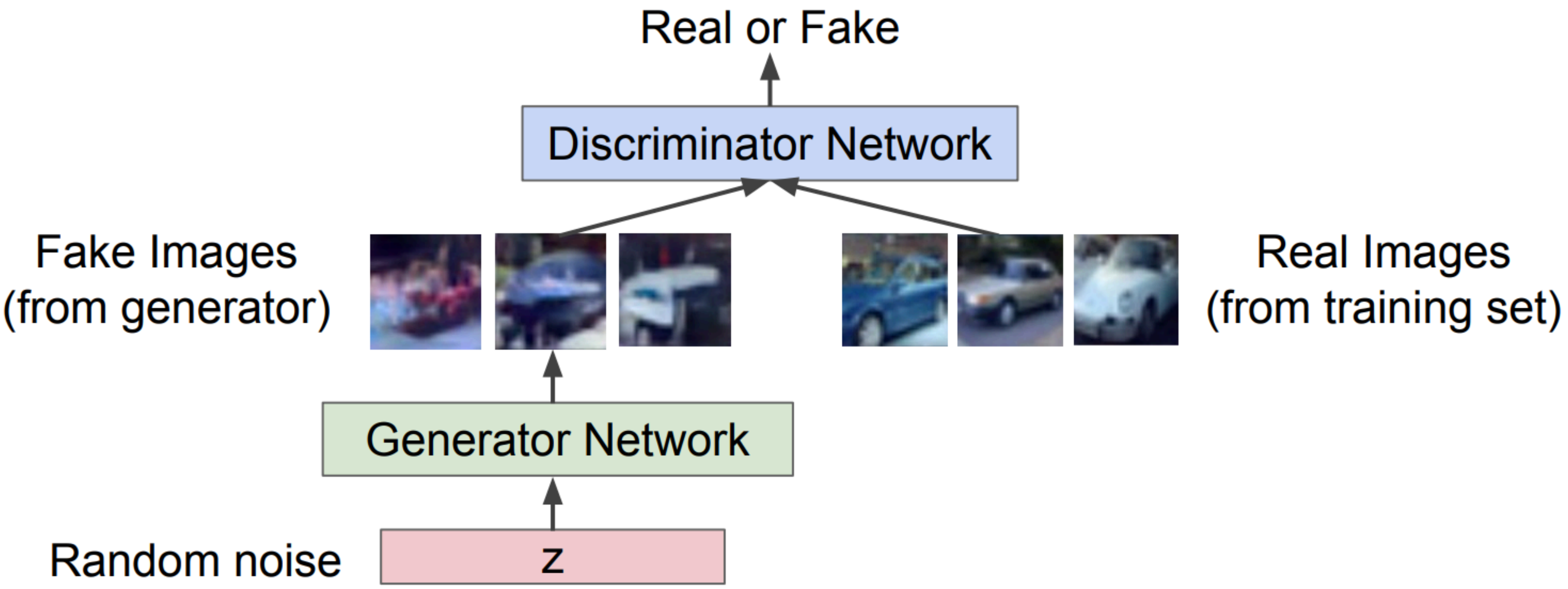


Suffered from blurry issues.

Why? Loss function (L2).

GAN

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images



Credit: Stanford CS231N

Training GAN

Train jointly in **minimax game**

Minimax objective function:

Discriminator outputs likelihood in (0,1) of real image

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log \underbrace{D_{\theta_d}(x)}_{\text{Discriminator output for real data x}} + \mathbb{E}_{z \sim p(z)} \log(1 - \underbrace{D_{\theta_d}(G_{\theta_g}(z))}_{\text{Discriminator output for generated fake data G(z)}}) \right]$$

- Discriminator (θ_d) wants to **maximize objective** such that D(x) is close to 1 (real) and D(G(z)) is close to 0 (fake)
- Generator (θ_g) wants to **minimize objective** such that D(G(z)) is close to 1 (discriminator is fooled into thinking generated G(z) is real)

Credit: Stanford CS231N

Voxel GAN

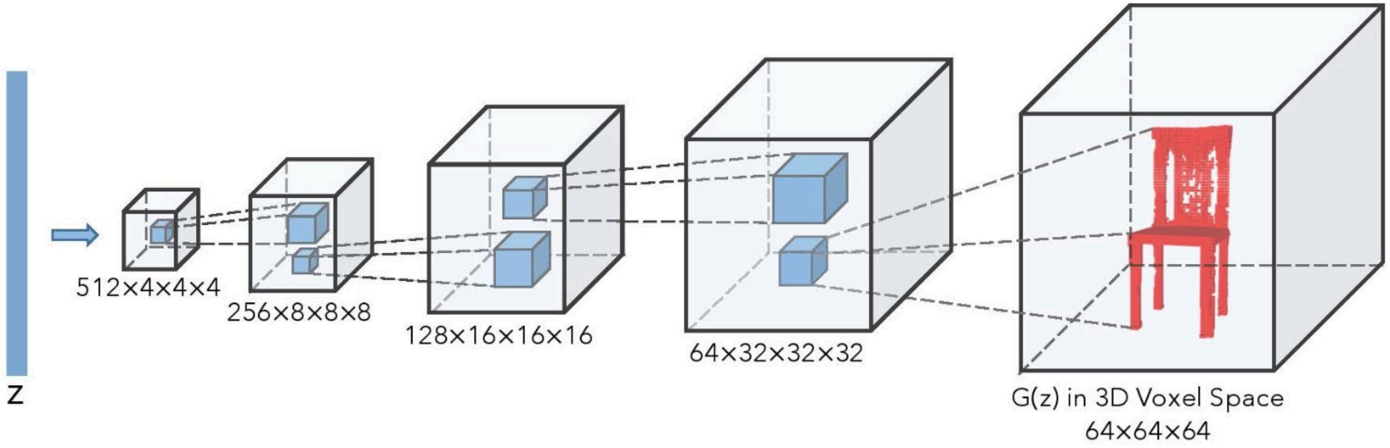
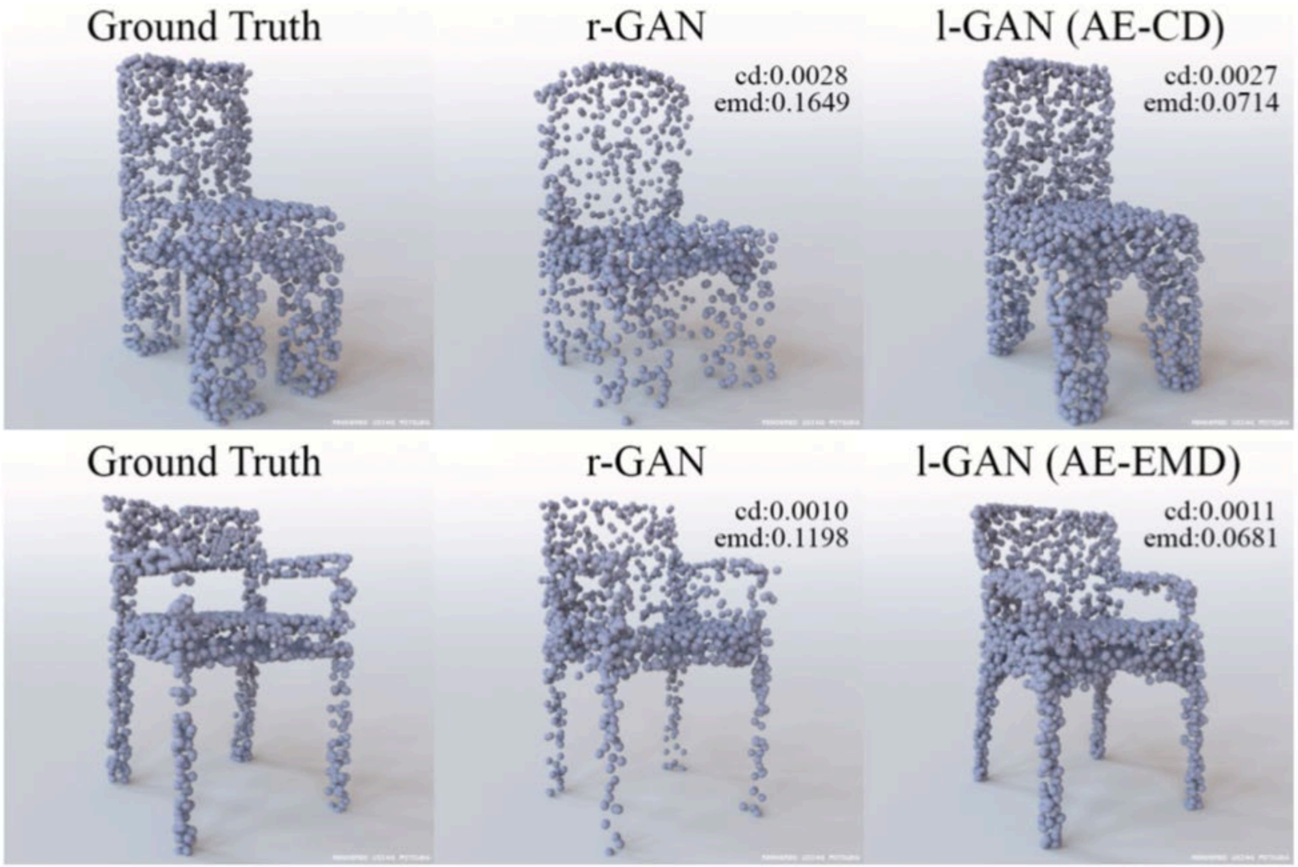


Figure 1: The generator of 3D Generative Adversarial Networks (3D-GAN)

Figure 2: Shapes synthesized by 3D-GAN

Wu et. al., **Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling**, NeurIPS 2016

Point Cloud GANs



ICML, 2018, Learning Representations and Generative Models for 3D Point Clouds, Panos Achlioptas, et. al.

More Applications

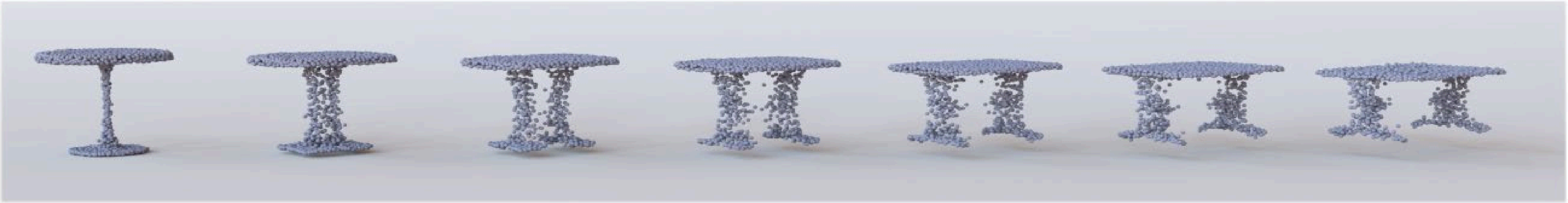


Figure 2. Interpolating between different point clouds, using our latent space representation. More examples for furniture and *human-form* objects (Bogo et al., 2017) are demonstrated in the Appendix in Figures 11 and 14, respectively.

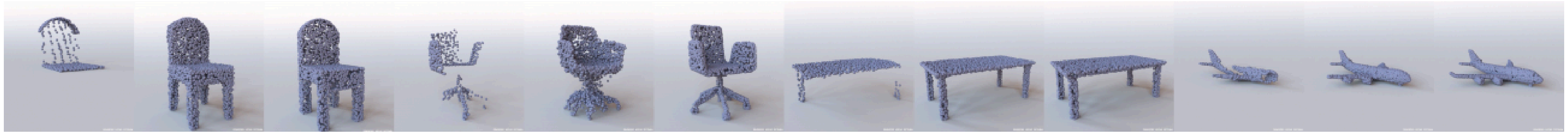
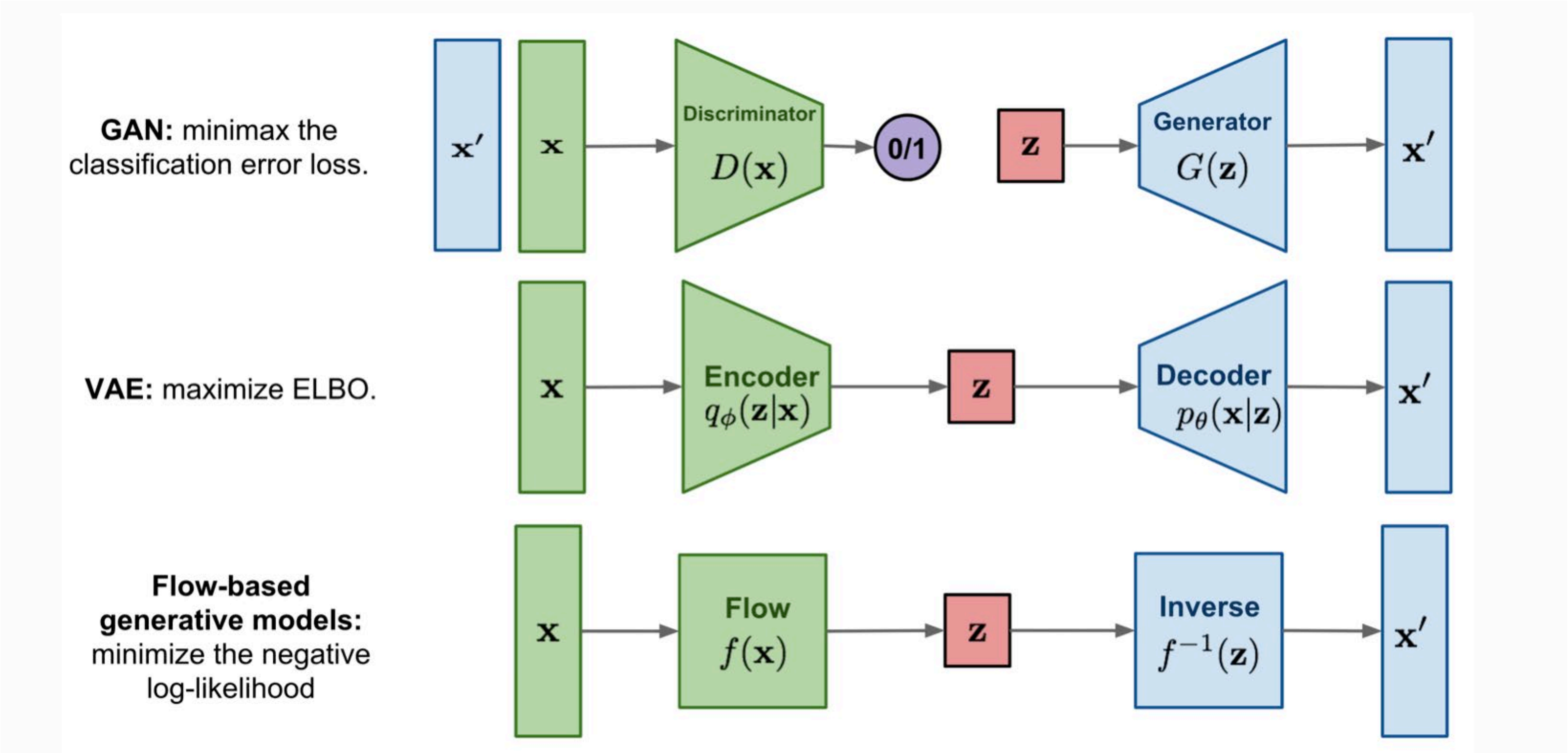


Figure 4. Point cloud *completions* of a network trained with partial and complete (input/output) point clouds and the EMD loss. Each triplet shows the partial input from the test split (left-most), followed by the network's output (middle) and the complete ground-truth (right-most).

ICML, 2018, Learning Representations and Generative Models for 3D Point Clouds, Panos Achlioptas, et. al.

Flow-based Generative Model

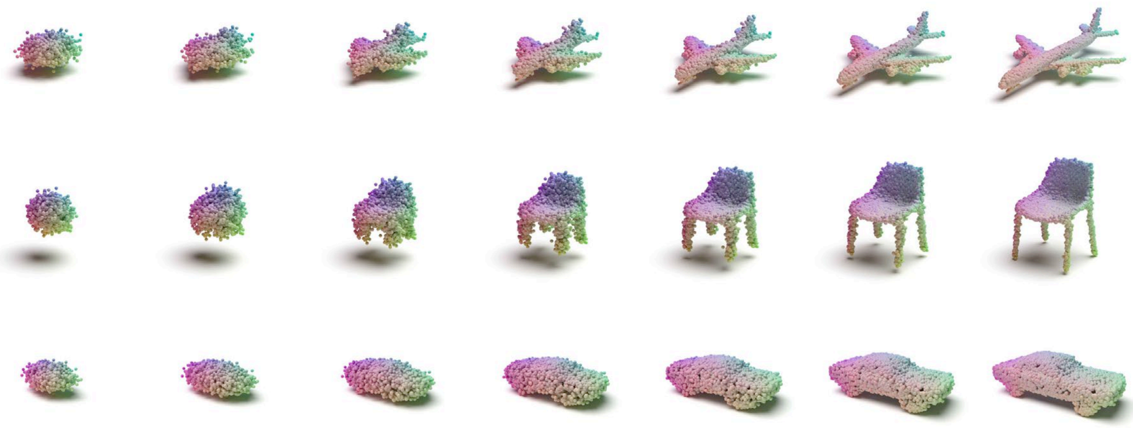


Flow-based model is constructed by a sequence of **invertible transformations.**
Explicitly modeling probability.   Loss: negative loglikelihood of z = f(x)

Image credit: [Lil'log]

Flow-based 3D Generative Model

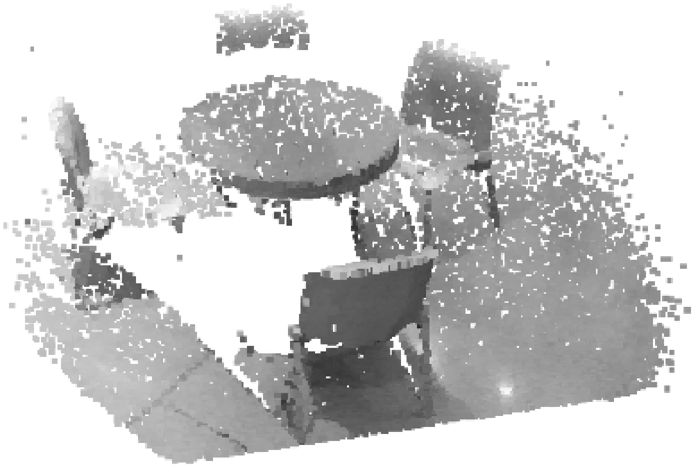
Discrete Point Flow Networks
From Univ. Grenoble Alpes



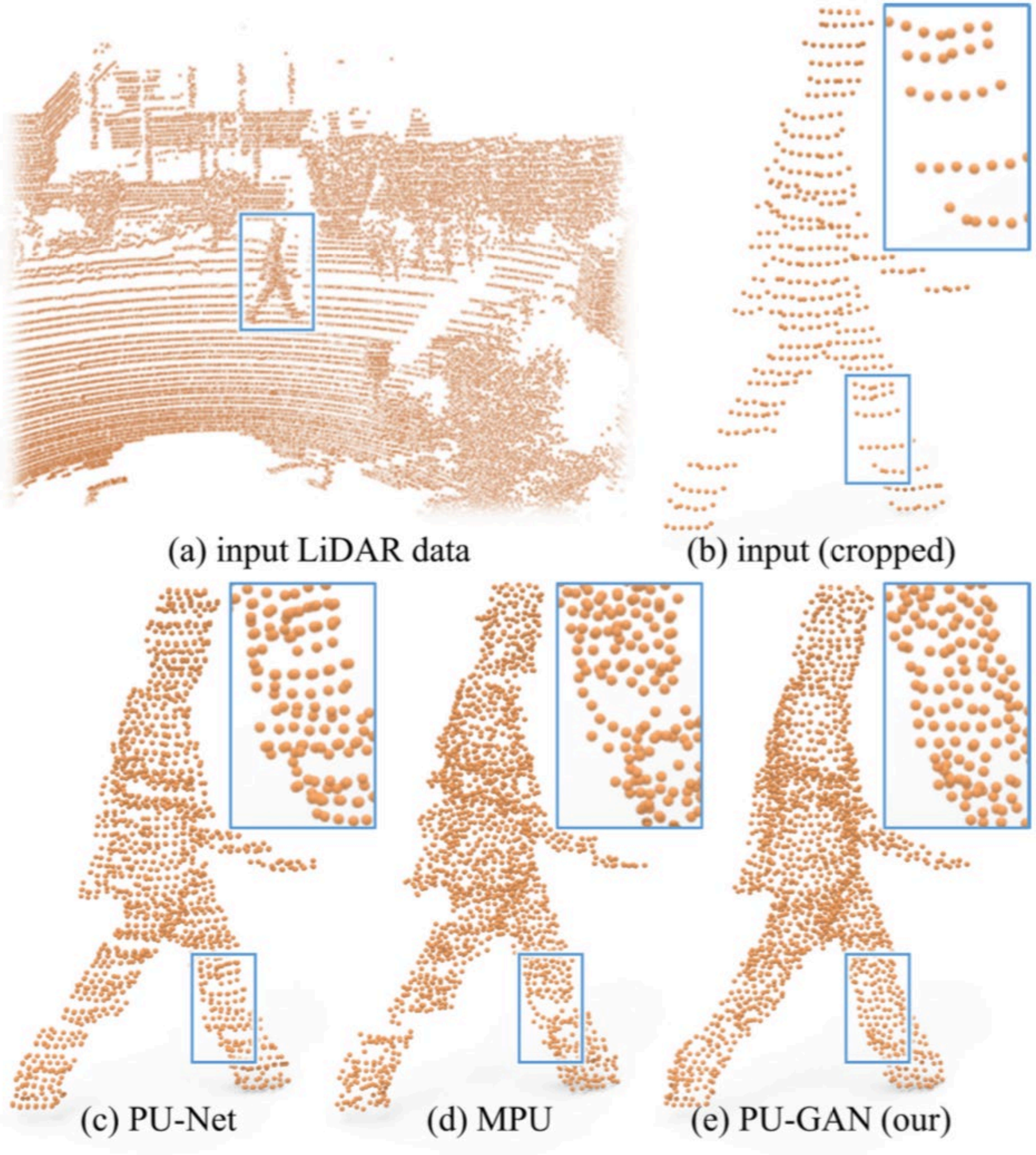
PointFlow (continuous normalizing flow)
From Cornell

Note that bijectivity requires same dimenisionality.
From left to right: latent points to generated points

Deep 3D Conditional Generative Models: CVAE, CGAN

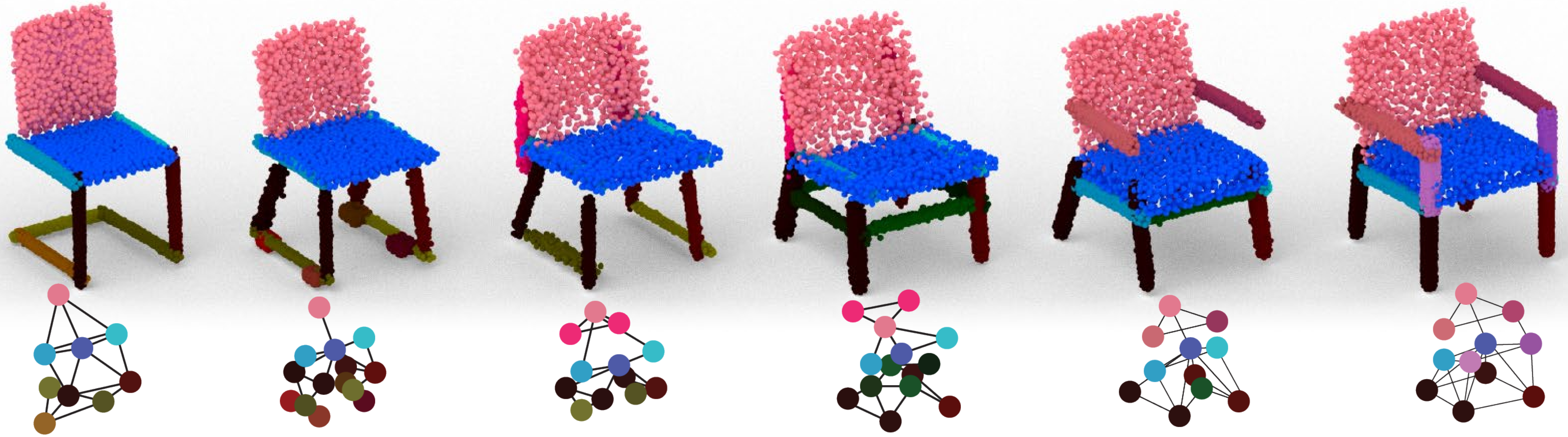


Point Cloud Upsampling



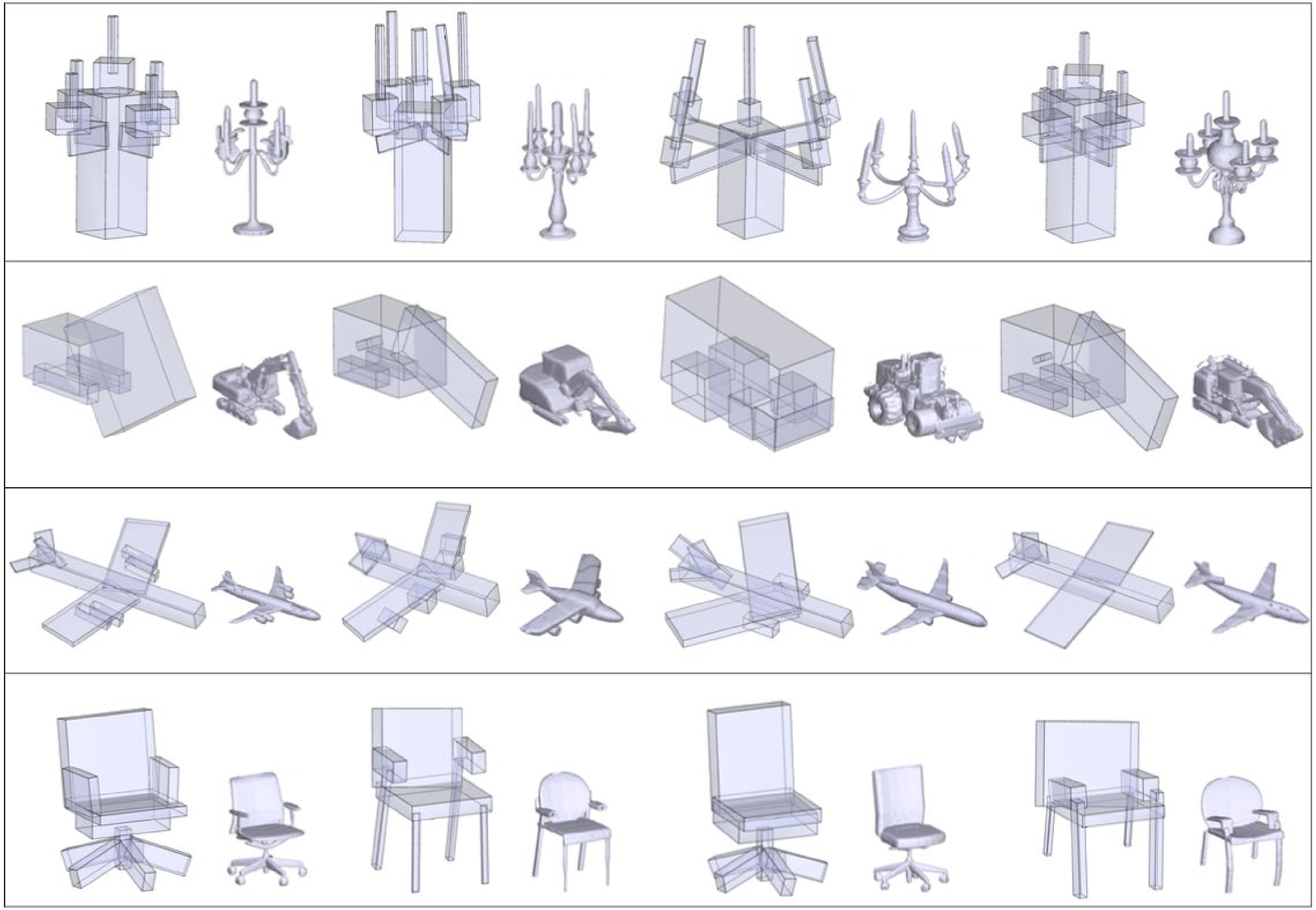
PU-GAN: a Point Cloud Upsampling Adversarial Network

StructureNet

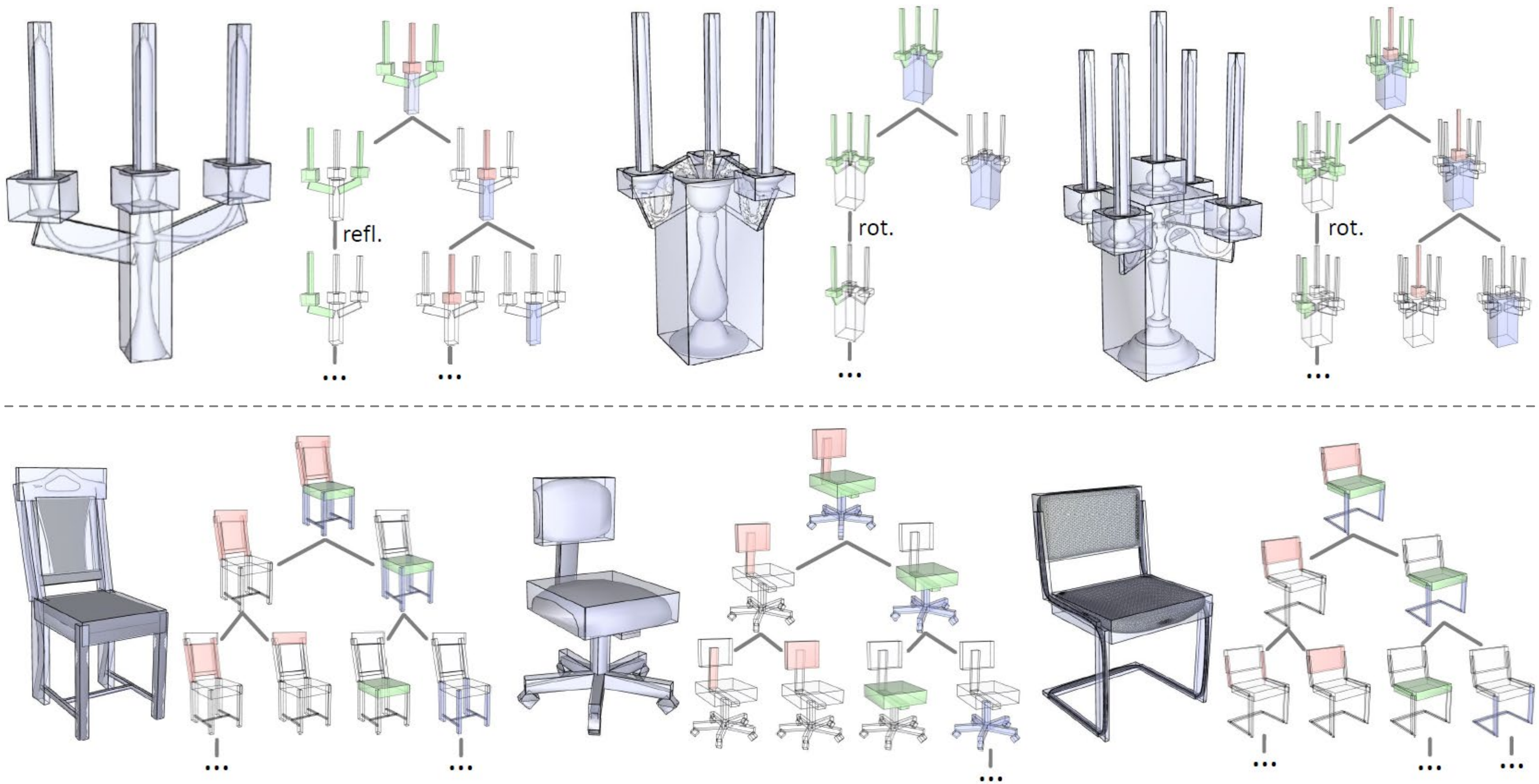


StructureNet: Hierarchical Graph Networks for 3D Shape Generation

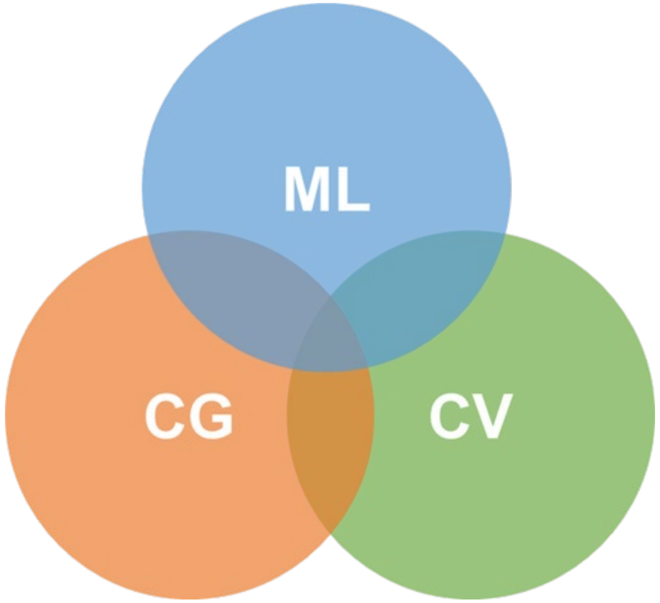
GRASS: Shape Synthesis



GRASS: Inferring Consistent Hierarchy



Thank You



Machine learning
Computer vision
Computer graphics