



# Robot Perception

## Segmentation & 3D Deep Learning

Dr. Felix Juefei Xu and Dr. Chen Feng

[cfeng@nyu.edu](mailto:cfeng@nyu.edu)

ROB-GY 6203, Fall 2022



# Overview

---

- Segmentation
  - + Semantic Segmentation
  - + Instance Segmentation
  - + Panoptic Segmentation
- 3D Deep Learning
  - + MVCNN
  - + VoxNet
  - + PointNet
  - + FoldingNet
  - + DeepMapping

\*: know how to code

++: know how to derive

+: know the concept



# References

---

- Sz: Ch 6.4
- Maturana, Daniel, and Sebastian Scherer. "Voxnet: A 3d convolutional neural network for real-time object recognition." *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2015.
- Su, Hang, et al. "Multi-view convolutional neural networks for 3d shape recognition." *Proceedings of the IEEE international conference on computer vision*. 2015.
- Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 652-660).
- Yang, Y., Feng, C., Shen, Y., & Tian, D. (2018). Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 206-215).
- Ding, Li, and Chen Feng. "DeepMapping: Unsupervised map estimation from multiple point clouds." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019.



# Semantic Segmentation: The Problem

## Classification



CAT

No spatial extent

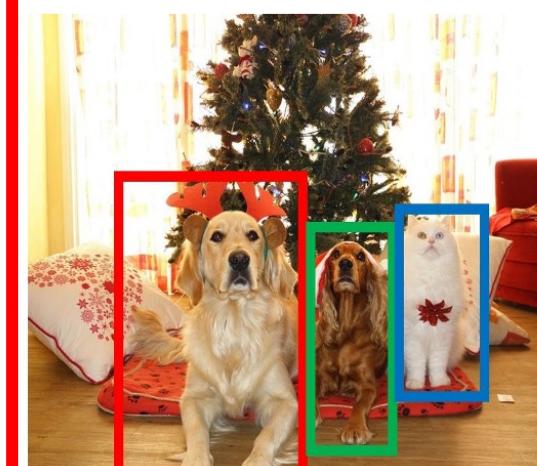
## Semantic Segmentation



GRASS, CAT,  
TREE, SKY

No objects, just pixels

## Object Detection



DOG, DOG, CAT

Multiple Object



DOG, DOG, CAT

## Instance Segmentation

[This image is CC0 public domain](#)

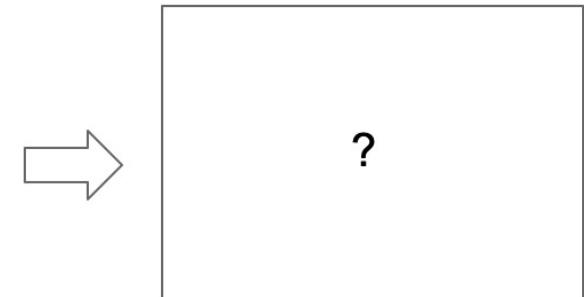


# Semantic Segmentation: The Problem



**GRASS, CAT,  
TREE, SKY, ...**

Paired training data: for each training image,  
each pixel is labeled with a semantic category.



At test time, classify each pixel of a new image.



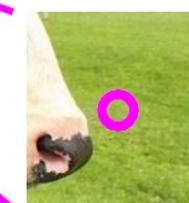
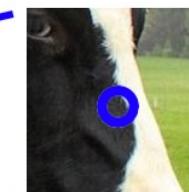
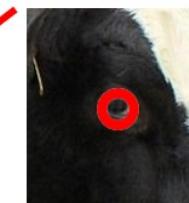
# Semantic Segmentation Idea: Sliding Window

Problem: Very inefficient! Not reusing shared features between overlapping patches

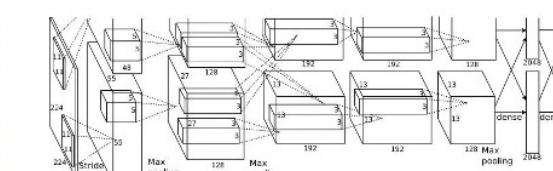
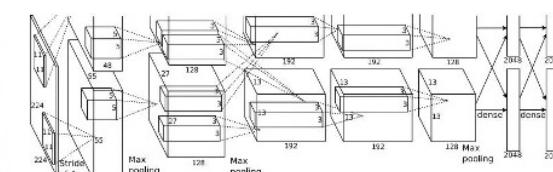
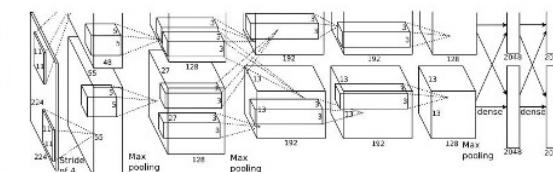
Full image



Extract patch



Classify center pixel with CNN



Classifying the image pixel by pixel is impossible without context.  
Processing small patches give some context for classifying

Cow

Cow

Grass

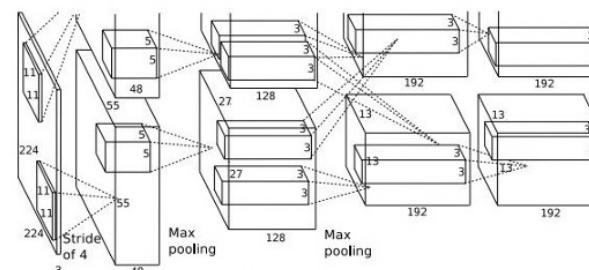
Farabet et al, "Learning Hierarchical Features for Scene Labeling," TPAMI 2013

Pinheiro and Collobert, "Recurrent Convolutional Neural Networks for Scene Labeling", ICML 2014



# Semantic Segmentation Idea: Convolution

Full image



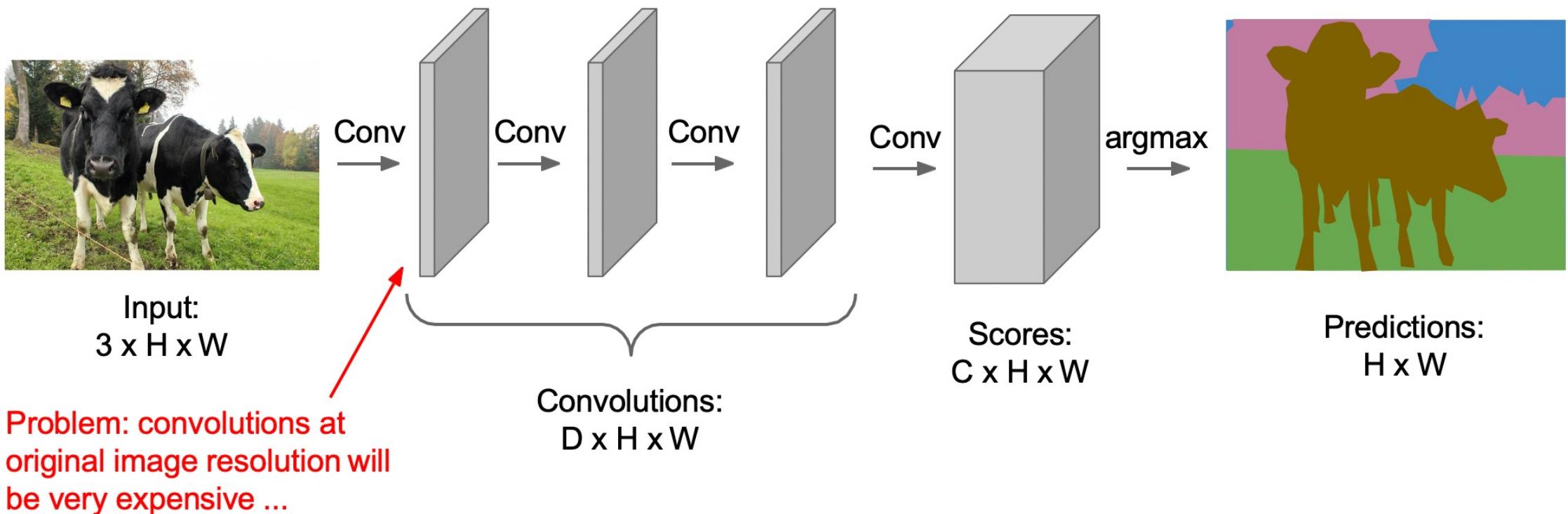
An intuitive idea: encode the entire image with conv net, and do semantic segmentation on top.

Problem: classification architectures often reduce feature spatial sizes to go deeper, but semantic segmentation requires the output size to be the same as input size.



# Semantic Segmentation Idea: Convolution

Design a network with only convolutional layers without downsampling operators to make predictions for pixels all at once!





# Semantic Segmentation Idea: Fully Convolutional

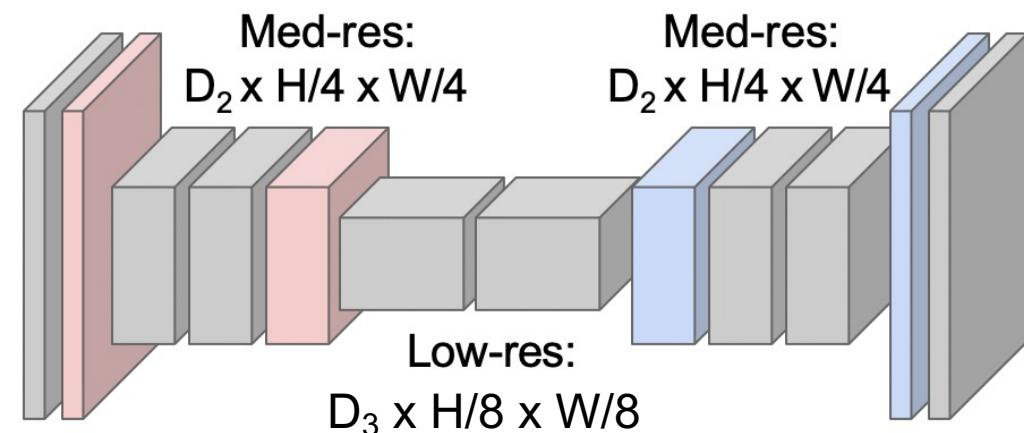
**Downsampling:**  
Pooling, strided convolution



Input:  
 $3 \times H \times W$

High-res:  
 $D_1 \times H/2 \times W/2$

Design network as a bunch of convolutional layers, with  
**downsampling** and **upsampling** inside the network!



**Upsampling:**  
???



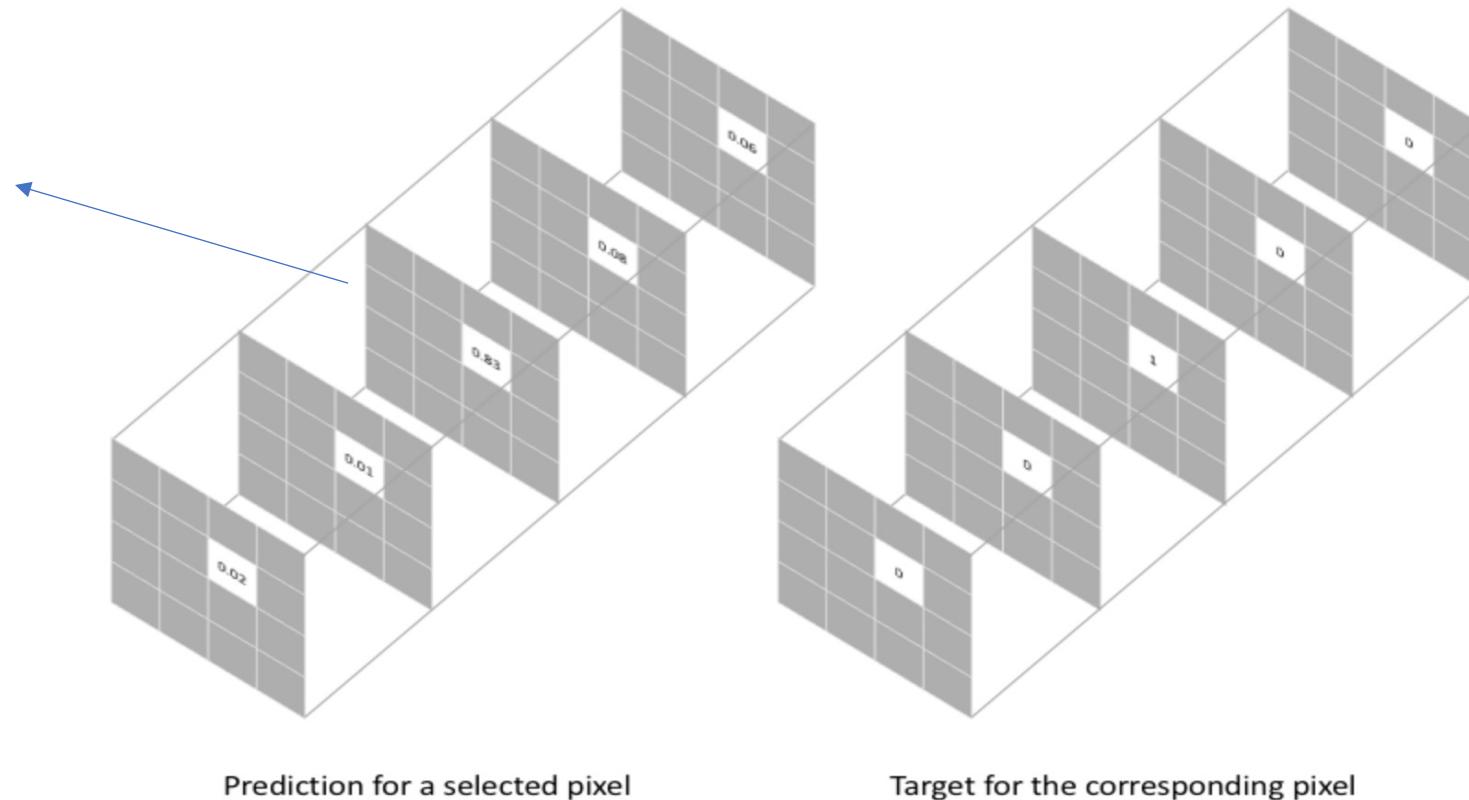
Long, Shelhamer, and Darrell, "Fully Convolutional Networks for Semantic Segmentation", CVPR 2015  
Noh et al, "Learning Deconvolution Network for Semantic Segmentation", ICCV 2015



# Semantic Segmentation Idea : Loss

- The loss function is basically the sum of pixel-wise multinomial logistic loss or cross entropy value over all the pixels.

C+1  
channels for  
each pixel  
(including  
background)



Pixel-wise loss is calculated as the log loss, summed over all possible classes

$$-\sum_{\text{classes}} y_{\text{true}} \log(y_{\text{pred}})$$

This scoring is repeated over all pixels and averaged



# Upsampling – Max Unpooling

## Max Pooling

Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

Input: 4 x 4

5	6
7	8

Output: 2 x 2

## Max Unpooling

Use positions from pooling layer

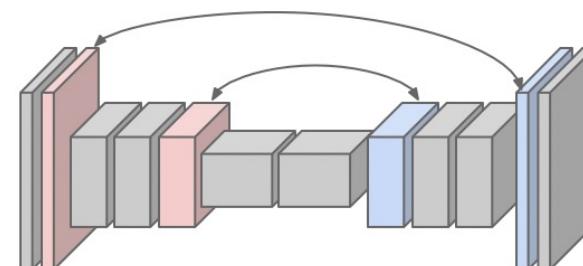
1	2
3	4

Input: 2 x 2

0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

Output: 4 x 4

Corresponding pairs of  
downsampling and  
upsampling layers



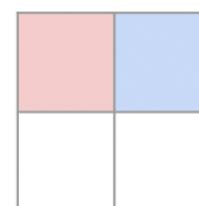


# Upsampling – Transposed Convolution

**Other names:**

- Deconvolution (bad)
- Upconvolution
- Fractionally strided convolution
- Backward strided convolution

**Q: Why is it called transpose convolution?**

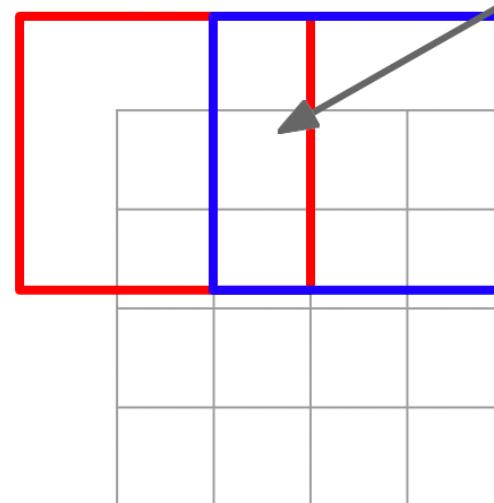


Input: 2 x 2

**3 x 3 transpose convolution, stride 2 pad 1**



Input gives  
weight for  
filter



Output: 4 x 4

Filter moves 2 pixels in  
the output for every one  
pixel in the input

Stride gives ratio between  
movement in output and  
input

Sum where  
output overlaps



# Semantic Segmentation Idea : Fully Convolutional

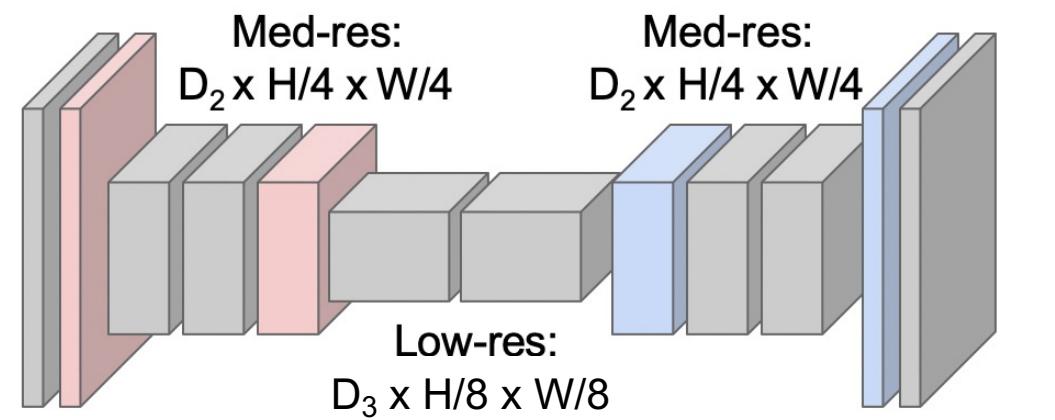
**Downsampling:**  
Pooling, strided  
convolution



Input:  
 $3 \times H \times W$

High-res:  
 $D_1 \times H/2 \times W/2$

Design network as a bunch of convolutional layers, with  
**downsampling** and **upsampling** inside the network!



**Upsampling:**  
Unpooling or strided  
transpose convolution



Predictions:  
 $H \times W$

Long, Shelhamer, and Darrell, "Fully Convolutional Networks for Semantic Segmentation", CVPR 2015  
Noh et al, "Learning Deconvolution Network for Semantic Segmentation", ICCV 2015



# Semantic Segmentation : Evaluation Metrics

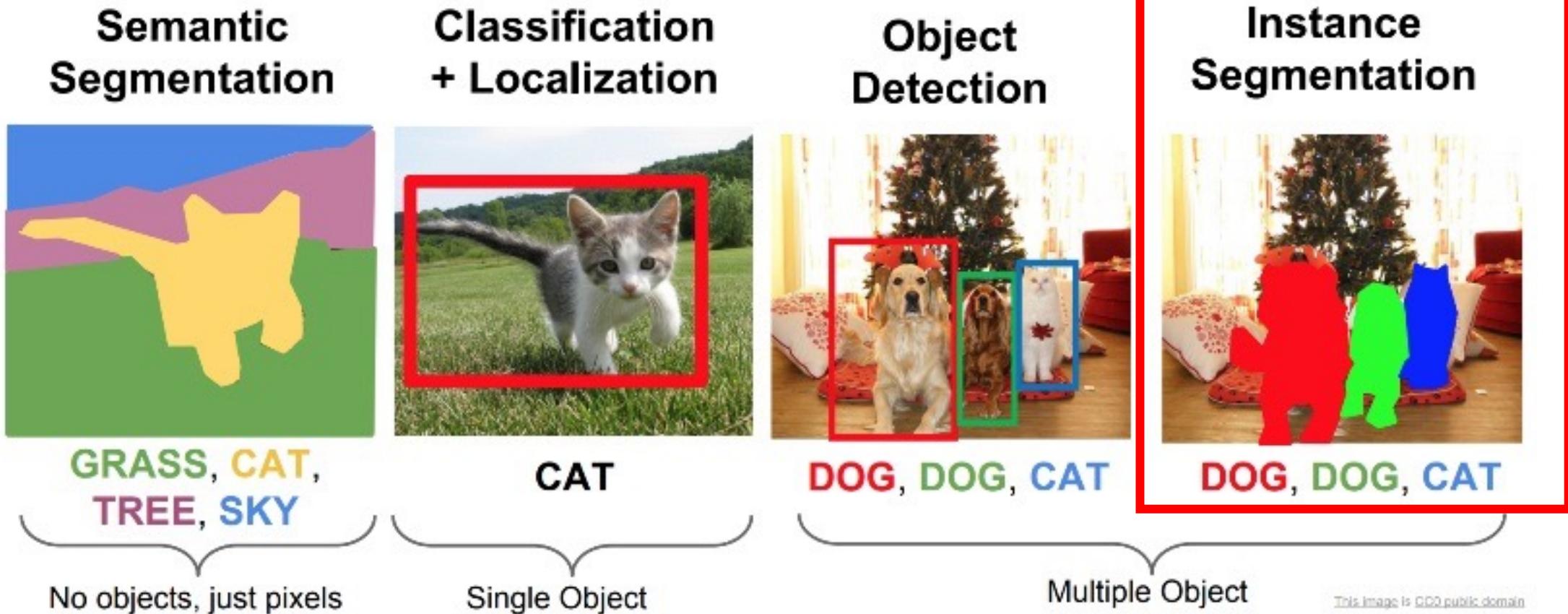
---

**Metrics** We report four metrics from common semantic segmentation and scene parsing evaluations that are variations on pixel accuracy and region intersection over union (IU). Let  $n_{ij}$  be the number of pixels of class  $i$  predicted to belong to class  $j$ , where there are  $n_{\text{cl}}$  different classes, and let  $t_i = \sum_j n_{ij}$  be the total number of pixels of class  $i$ . We compute:

- pixel accuracy:  $\sum_i n_{ii} / \sum_i t_i$
- mean accuracy:  $(1/n_{\text{cl}}) \sum_i n_{ii} / t_i$
- mean IU:  $(1/n_{\text{cl}}) \sum_i n_{ii} / \left( t_i + \sum_j n_{ji} - n_{ii} \right)$
- frequency weighted IU:  
$$(\sum_k t_k)^{-1} \sum_i t_i n_{ii} / \left( t_i + \sum_j n_{ji} - n_{ii} \right)$$



# Instance Segmentation



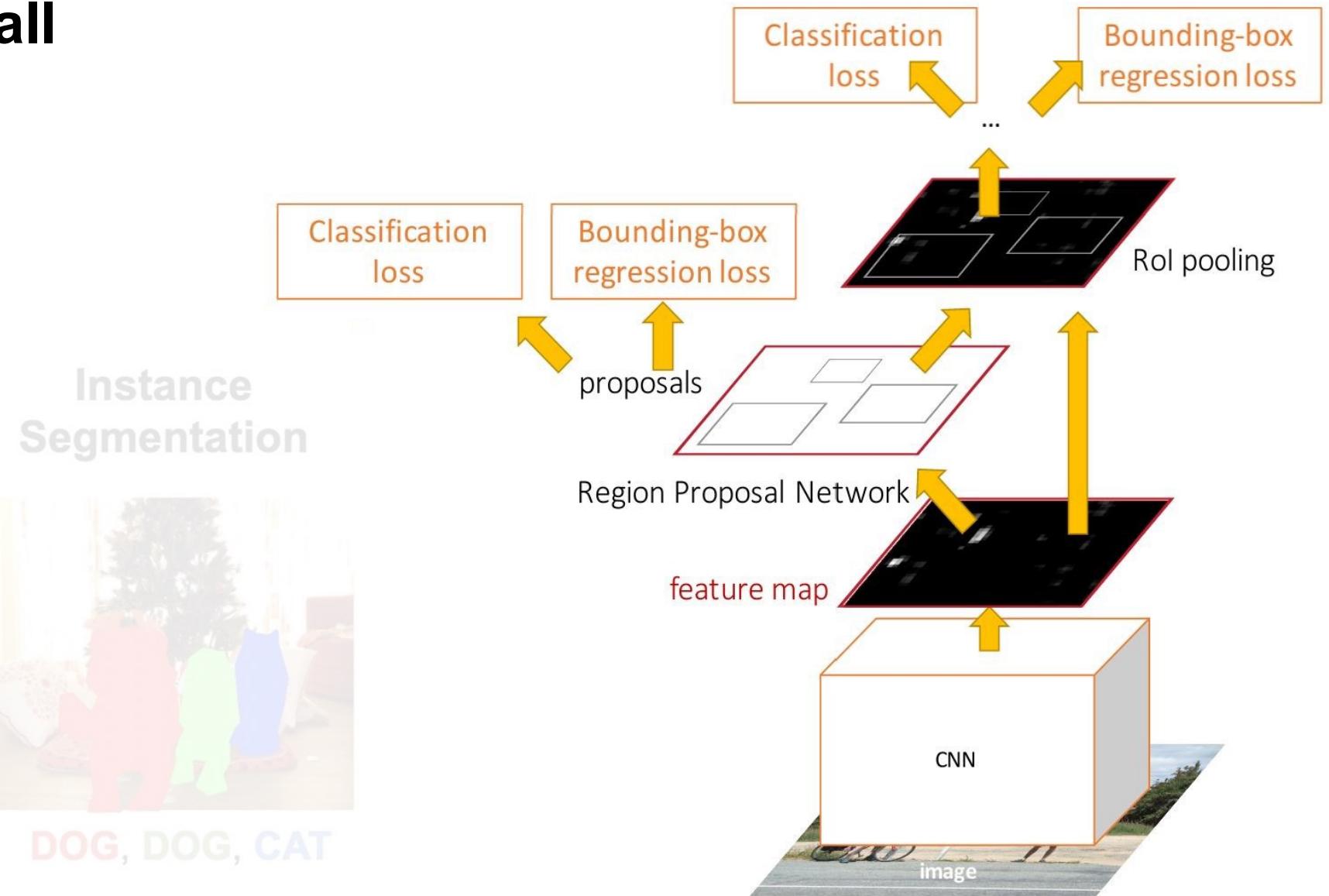


# Faster R-CNN- Recall

## Object Detection

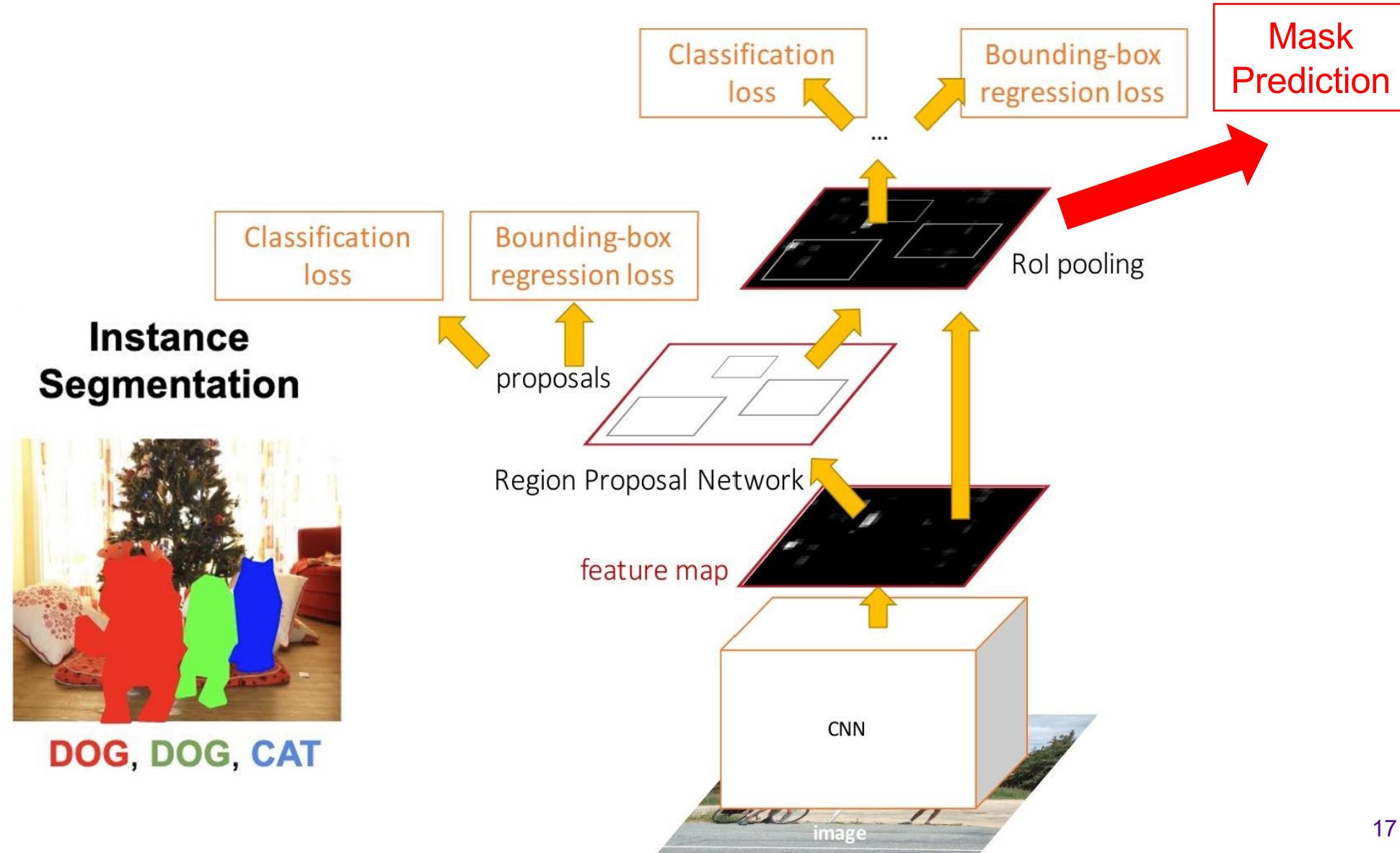
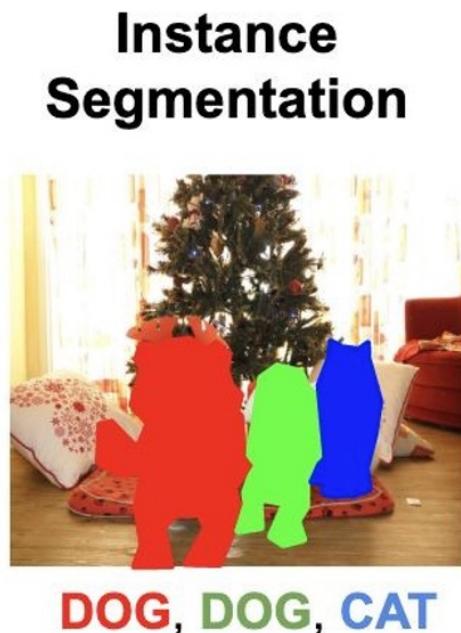


## Instance Segmentation





# Instance Segmentation – Mask R-CNN

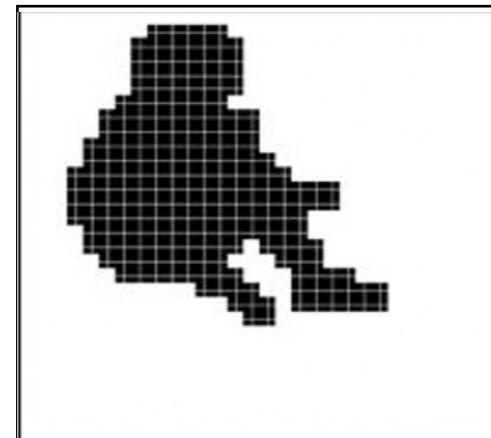




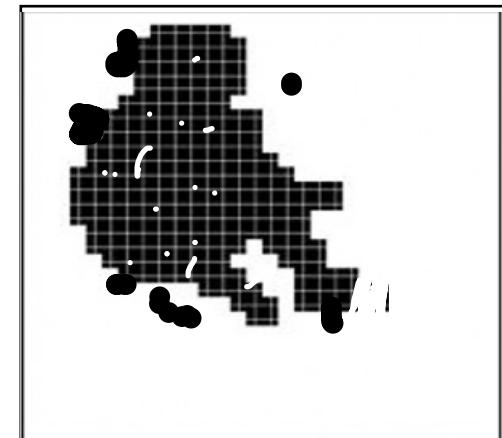
# Instance Segmentation – Mask R-CNN Loss

The loss is basically the average of pixel-wise binary cross entropy

Ground truth mask



Predicted mask



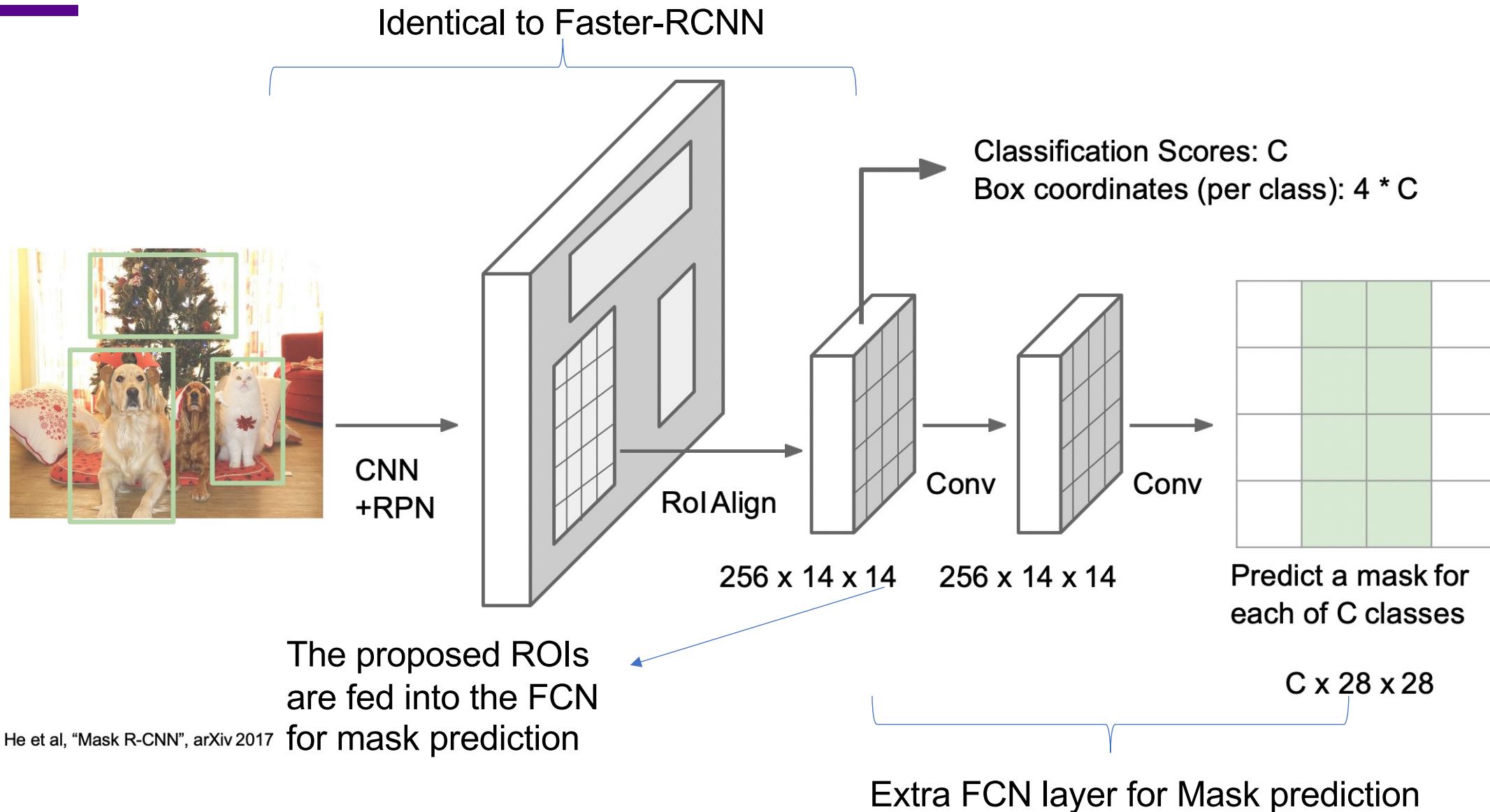
$$\mathcal{L}_{\text{mask}} = -\frac{1}{m^2} \sum_{1 \leq i, j \leq m} [y_{ij} \log \hat{y}_{ij}^k + (1 - y_{ij}) \log(1 - \hat{y}_{ij}^k)]$$

where  $y_{ij}$  is the label of a cell  $(i, j)$  in the true mask for the region of size  $m \times m$ ;  $\hat{y}_{ij}^k$  is the predicted value of the same cell in the mask learned for the ground-truth class  $k$ .

The remaining loss terms are exactly from Faster R-CNN (Lecture 10, Slide 78-79)

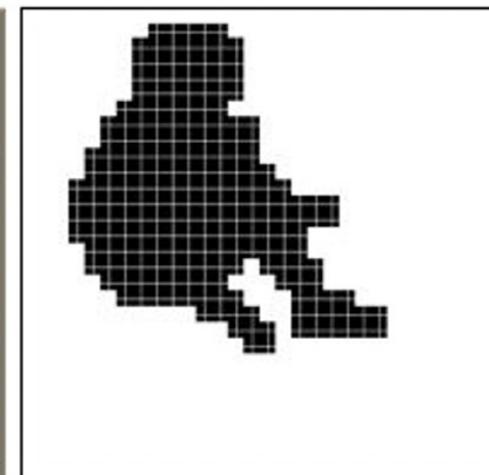
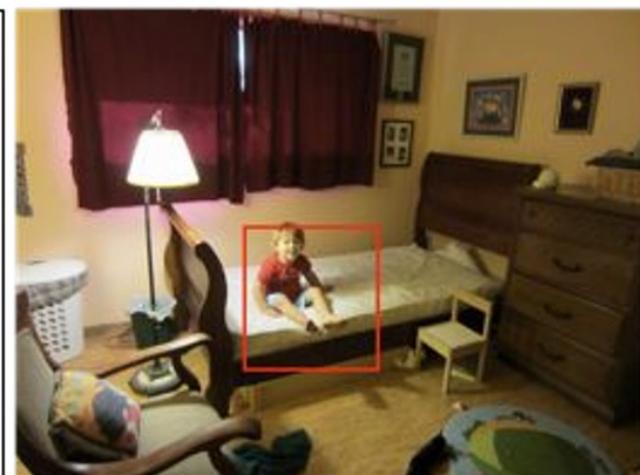
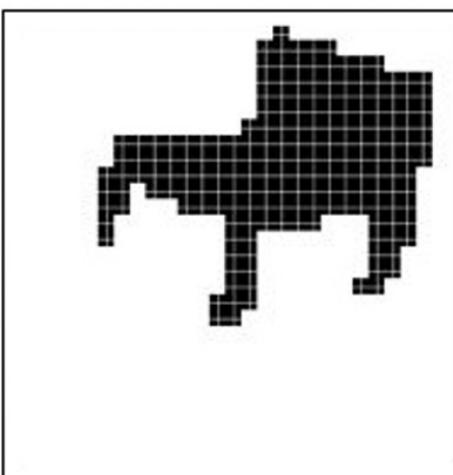
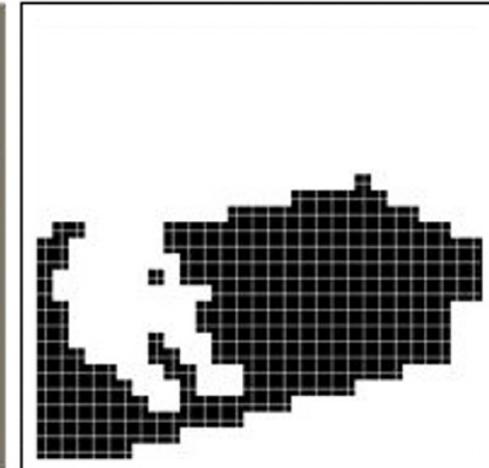
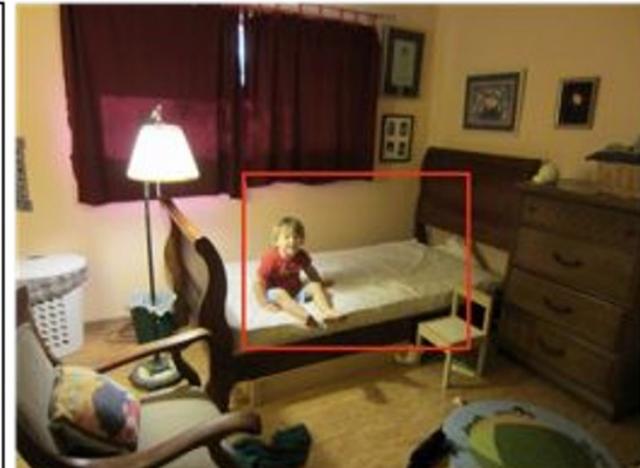
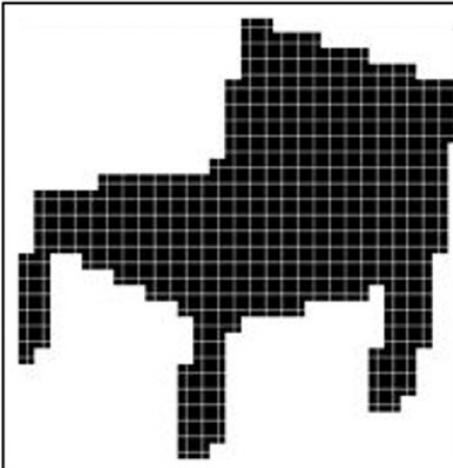


# Mask R-CNN : Architecture





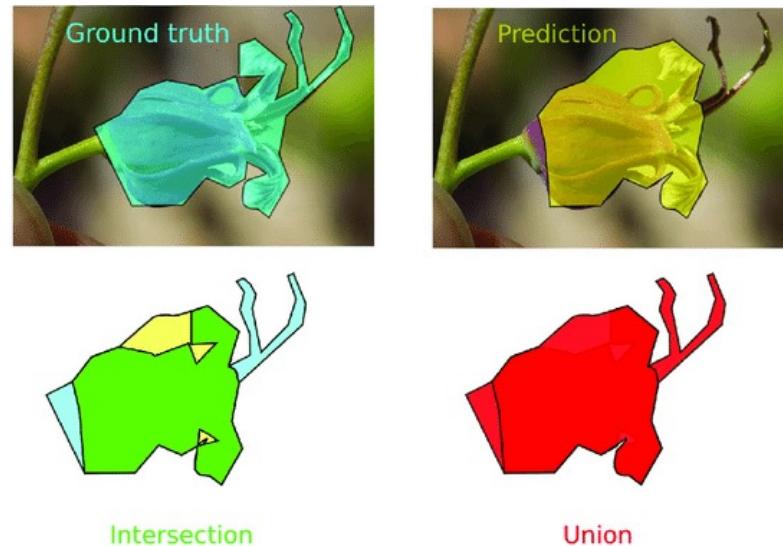
# Mask R-CNN: Example Mask Training Targets





# Mask R-CNN: Evaluation

- Mask IoU ( Intersection over Union )

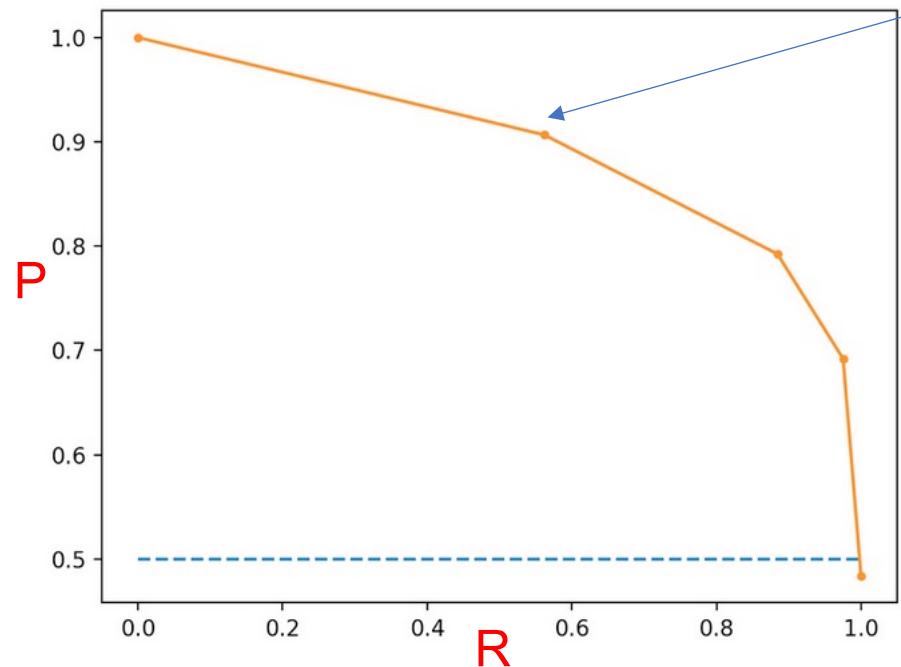


$$\text{IoU} = \frac{\text{Intersection}}{\text{Union}}$$

- mAP (Mean average precision)
  - averaged over categories and IoU thresholds

# Mean Average Precision

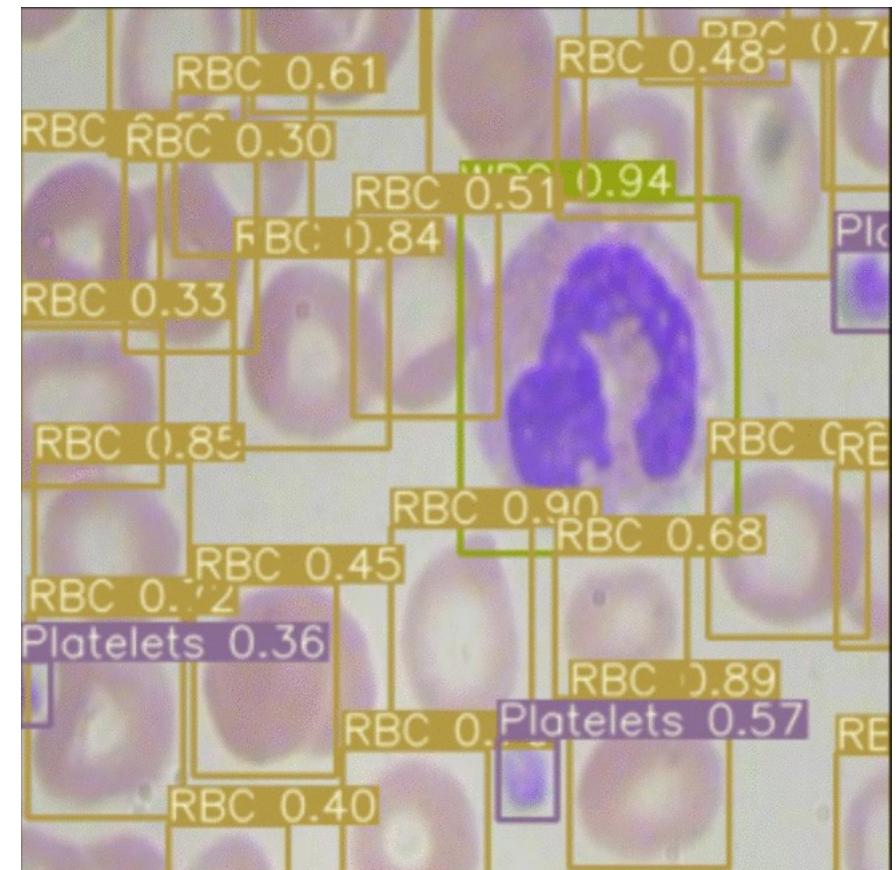
- First, draw precision-recall curve by varying confidence threshold



Average Precision can be computed by finding the Area Under the Curve (AUC) of the PR curve.

$$\text{AP} = \int_0^1 p(r)dr$$

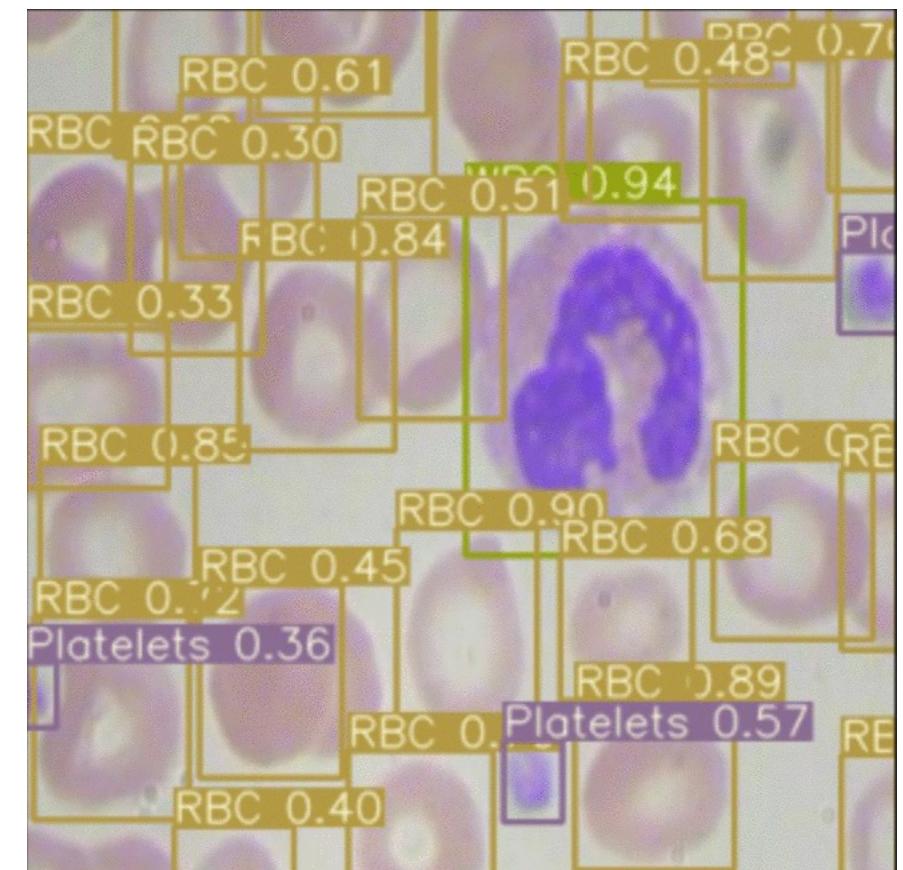
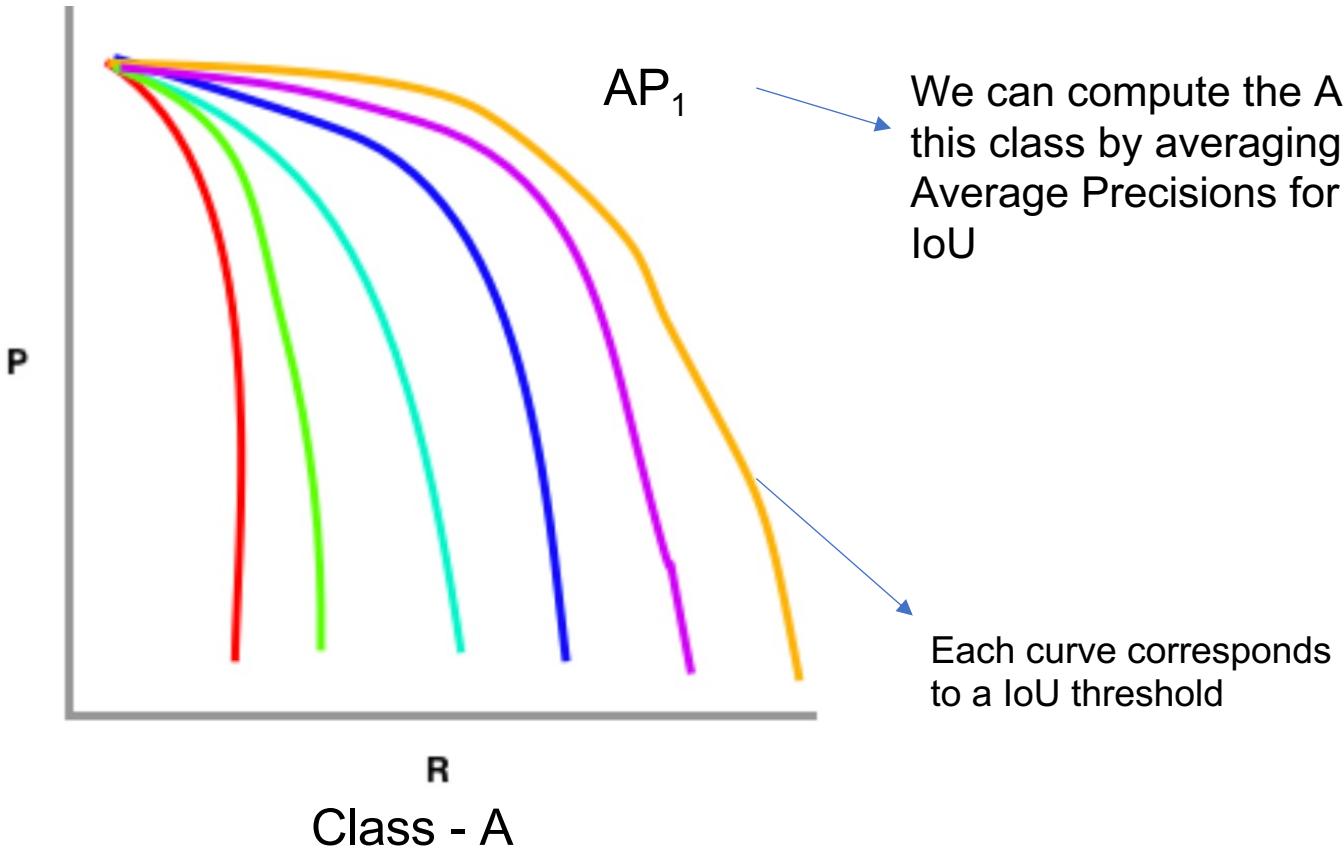
Precision and recall is plotted  
for each confidence threshold





# Mean Average Precision

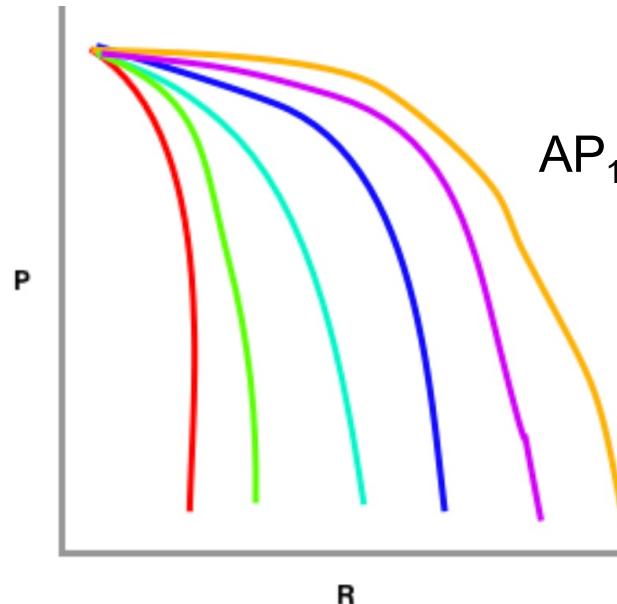
- The precision and recall values depend on the IoU threshold
- Secondly, for each class, draw multiple precision-recall curves, one for each IoU



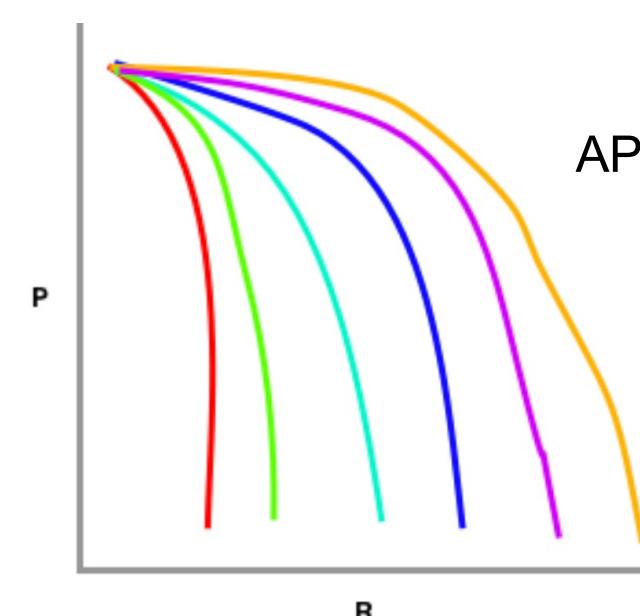


# Mean Average Precision

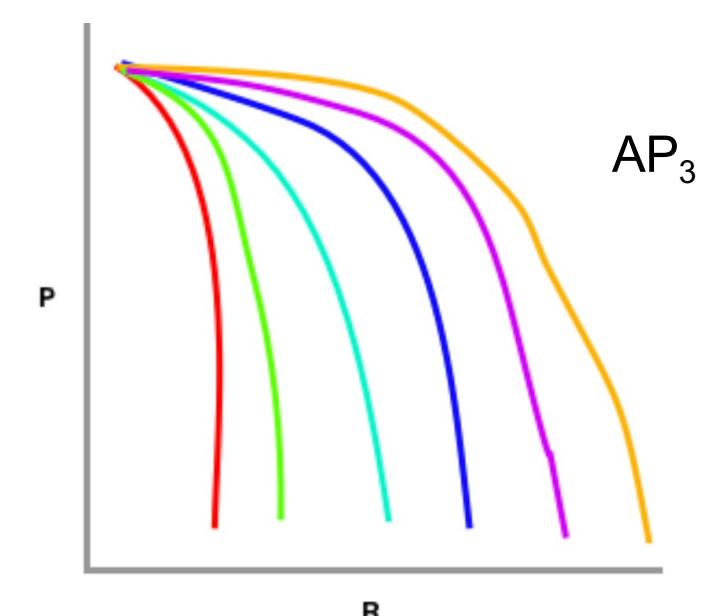
Class - A



Class - B



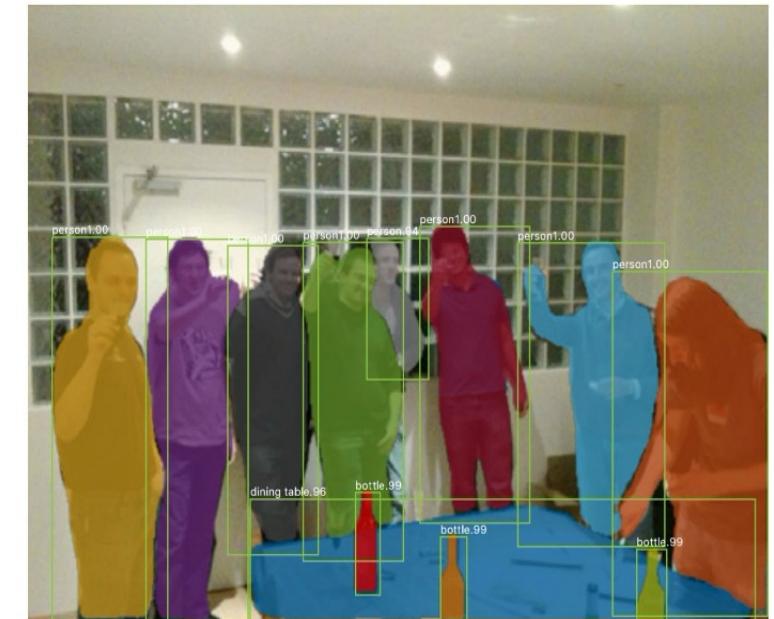
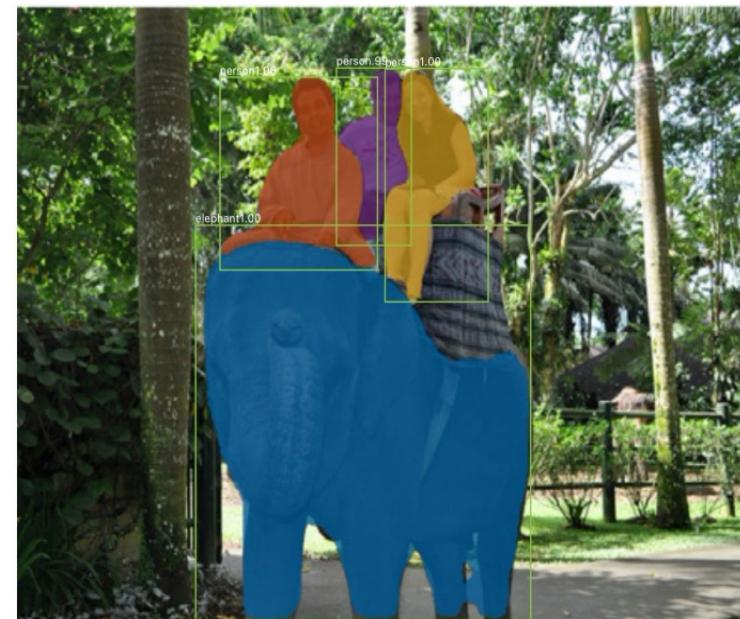
Class - C



Draw multiple Precision recall curves for each class

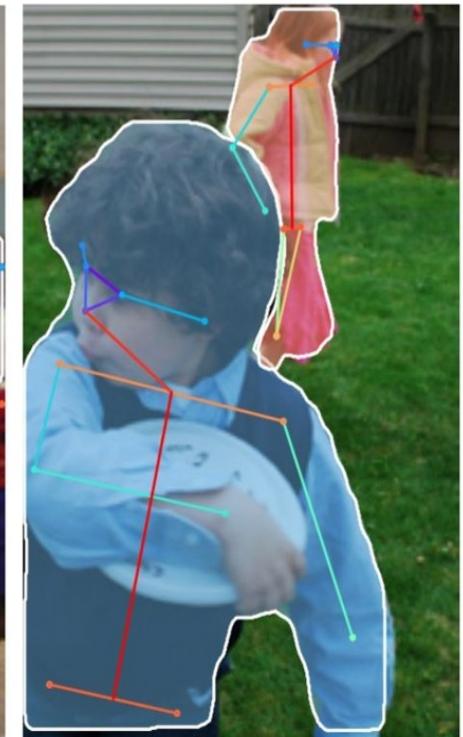
Mean Average Precision = Mean of Average Precisions of each class =  $1/3 (AP_1 + AP_2 + AP_3)$

# Mask R-CNN : Results





# Mask R-CNN : Human Pose Estimation





# Panoptic Segmentation : Task

- In semantic segmentation, the goal is to classify each pixel into the given classes.
- In instance segmentation, we care about segmentation of the instances of objects separately.
- The panoptic segmentation combines semantic and instance segmentation such that all **pixels are assigned a class label and all object instances are uniquely segmented**.

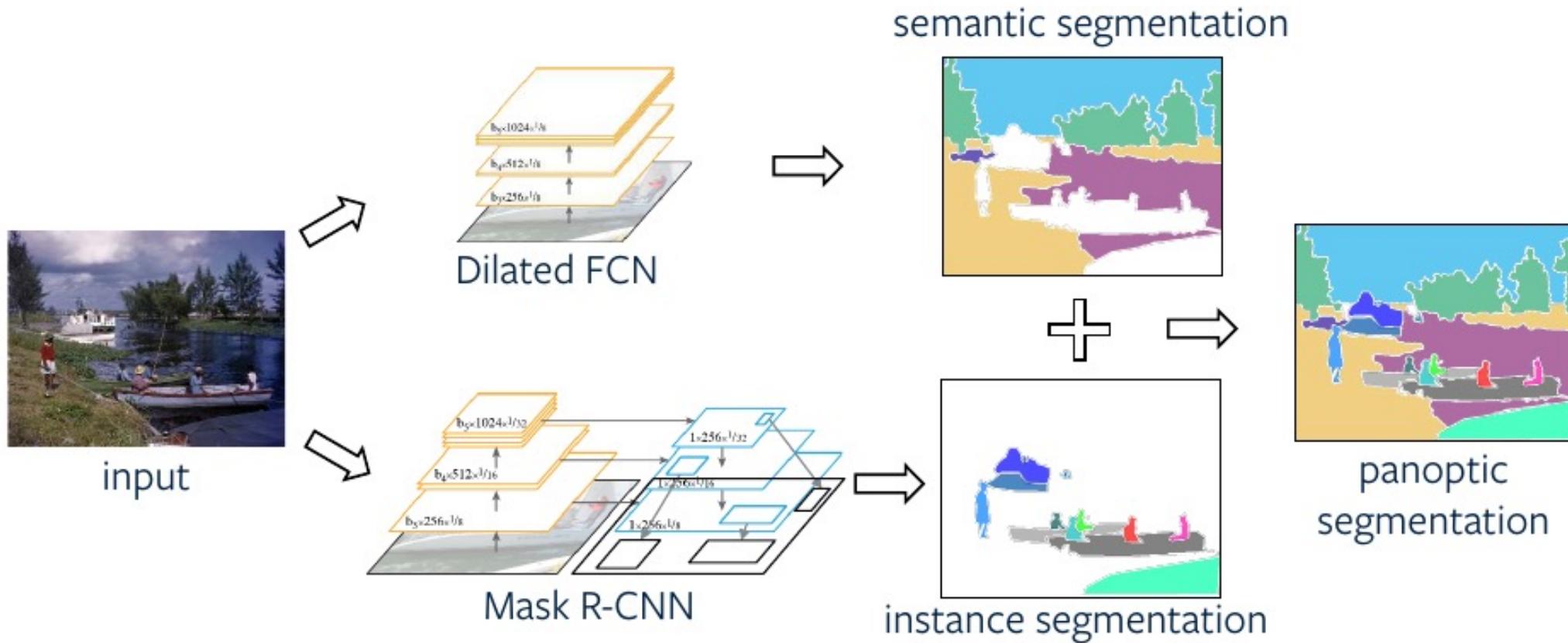


Left: semantic segmentation, middle: instance segmentation, right: panoptic segmentation



# Panoptic Segmentation : Naïve Model

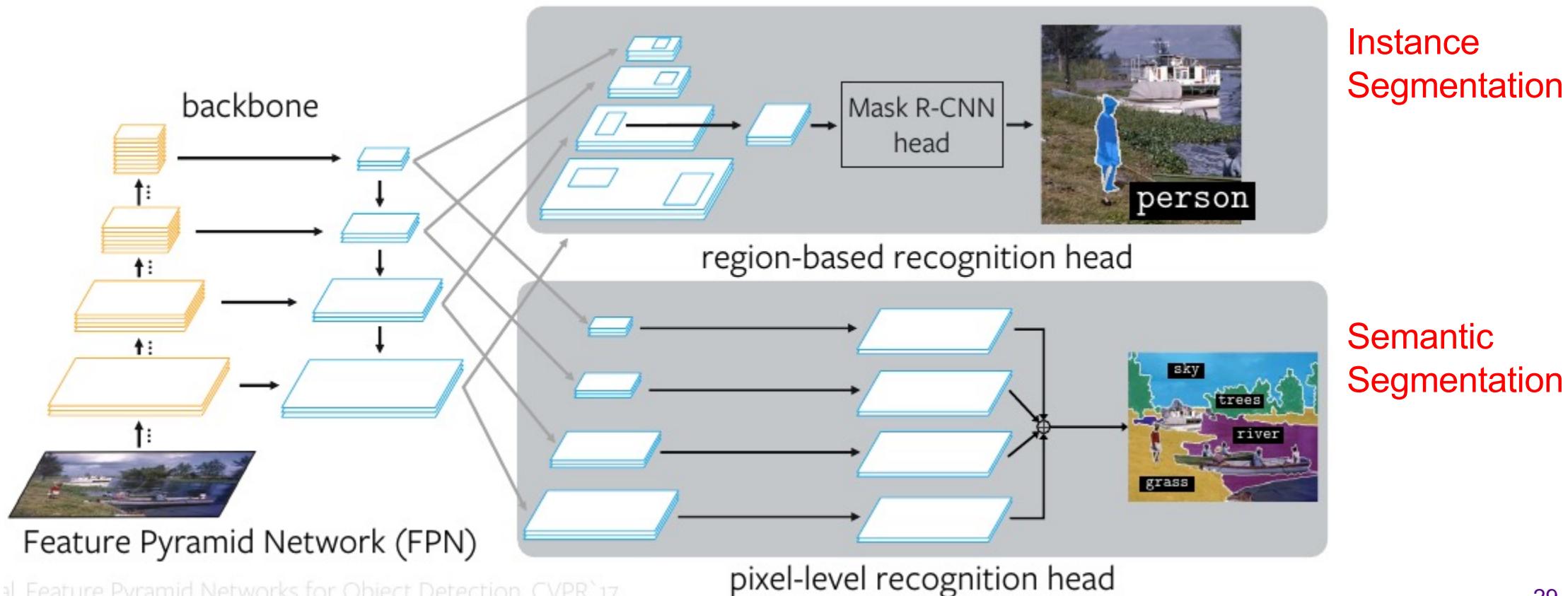
- One of the ways to solve the problem of panoptic segmentation is to **combine the predictions from semantic and instance segmentation models**
  - e.g. FCN and Mask R-CNN, to get panoptic predictions.





# Panoptic Segmentation: Panoptic FPN (Feature Pyramid Network)

- The idea is to use FPN for multi-level feature extraction as backbone, which is to be used for region-based instance segmentation as in case of Mask R-CNN, and add a parallel dense-prediction branch on top of same FPN features to perform semantic segmentation.





# Panoptic Segmentation : Panoptic FPN Training

- During training, the instance segmentation branch has three losses (**classification loss**), (**bounding-box loss**), and (**mask loss**). The semantic segmentation branch has semantic loss computed as the per-pixel cross-entropy between the predicted and the ground truth labels
- In addition, a weighted combination of the semantic and instance loss is used by adding two tuning parameters to get the panoptic loss.

$$L = \lambda_i(L_{cls} + L_{bbox} + L_{mask}) + \lambda_s L_s$$

- It is observed that the losses from these two branches have different scales and normalization policies. Simply adding them degrades the final performance for one of the tasks



# Panoptic Segmentation : Evaluation metrics

- Panoptic Quality (PQ)

$$PQ = \underbrace{\frac{\sum_{(p,g) \in TP} IoU(p, g)}{|TP|}}_{\text{segmentation quality (SQ)}} \times \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{\text{recognition quality (RQ)}}$$

- Mean IoU ( Intersection over Union )

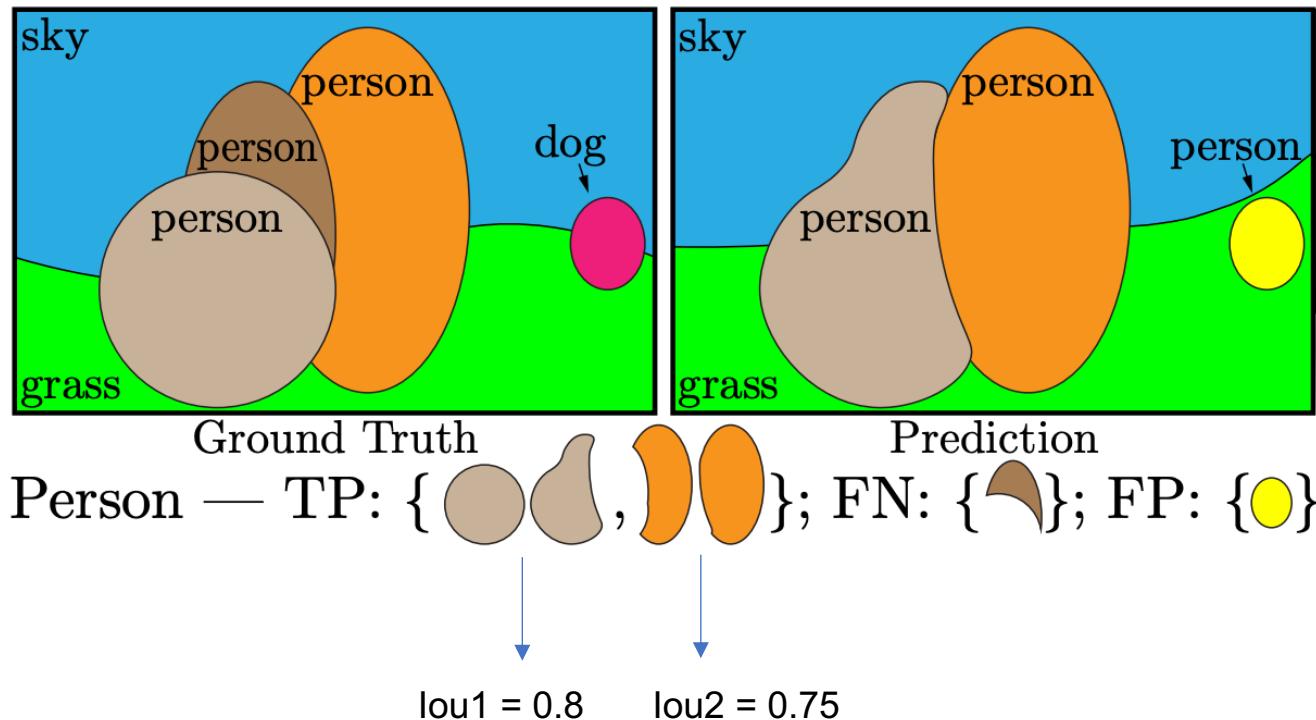
- Mean Average Precision

Segmentation quality ( SQ ) is evaluating how closely matched our segments are with their ground truths

This metric is a combination of precision and recall, attempting to identify how effective our model is at getting a prediction right



# Panoptic Quality - Example



$$SQ_{\{\text{Person}\}} = (\text{IoU}_1 + \text{IoU}_2) / 2 = 0.775$$

$$RQ_{\{\text{Person}\}} = 2 / (2 + 0.5 * 1 + 0.5 * 1) = 0.66$$

$$PQ_{\{\text{Person}\}} = SQ_{\{\text{Person}\}} * RQ_{\{\text{Person}\}} = 0.516$$

$$|TP|_{\{\text{Person}\}} = 2$$

$$|FP|_{\{\text{Person}\}} = 1$$

$$|FN|_{\{\text{Person}\}} = 1$$



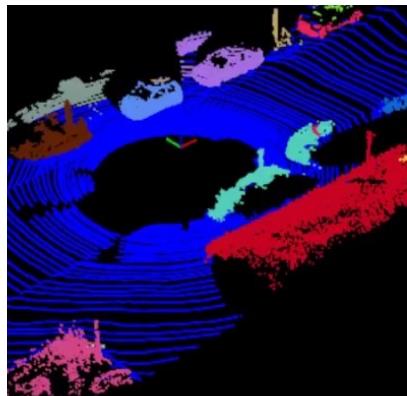
# 3D Deep Learning

- **Multi-view Convolutional Neural Networks for 3D Shape Recognition**
- **VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition**
- **PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation**
- **FoldingNet: Point Cloud Auto-encoder via Deep Grid Deformation**
- **DeepMapping: Unsupervised Map Estimation from Multiple Point Clouds**

# Why Deep Learning on 3D Data?

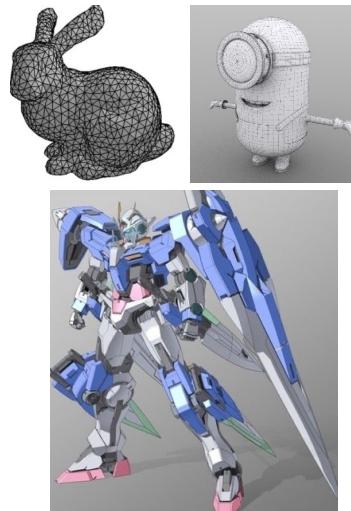
- How to enable robots to create maps and understand scenes in 3D?
- An important form of data – various application domains
- Intrinsically different than images – challenges for existing deep networks

Robotics



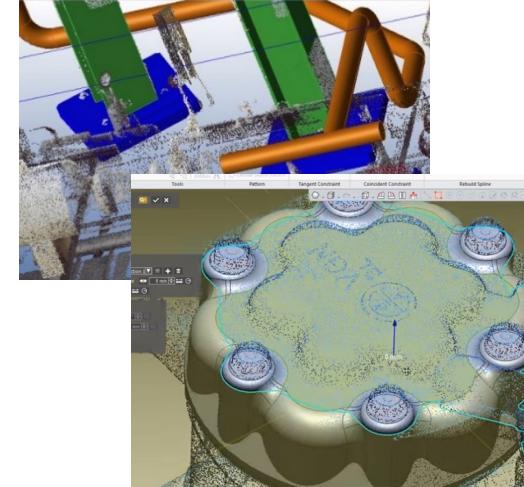
<https://www.youtube.com/watch?v=7NNpvtdrHkU>

Graphics/3DP



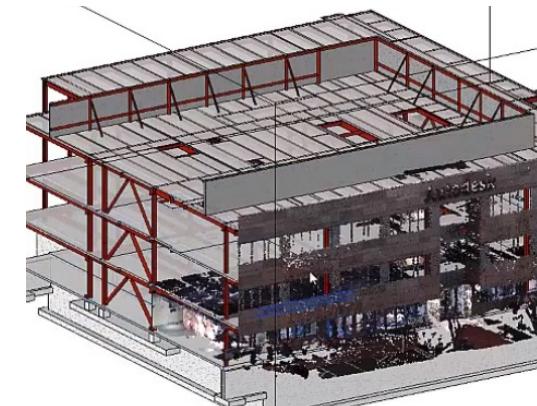
<https://www.pinterest.com/pin/134756213823244639/>

Mechanical Engineering



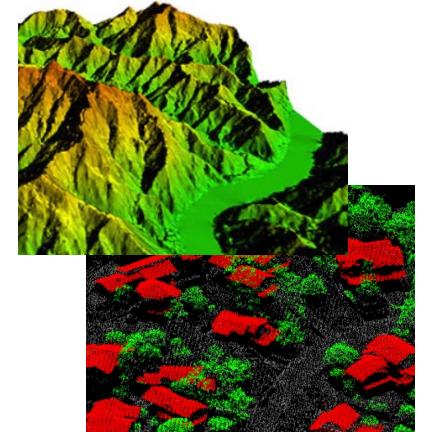
<https://www.youtube.com/watch?v=UD4asn3gkNI>

Civil Engineering



<https://www.youtube.com/watch?v=HhV6LAZ3DN0>

Geospatial Science



<http://www.aamgroup.com/services-and-technology/aerial-survey>



# 3D Input Representation

## Voxel

- ✓ 3D CNN
- Implicit representation
- ✗ Resolution/Scalability



[https://www.planetminecraft.com/  
project/giant-snowman-1638162/](https://www.planetminecraft.com/project/giant-snowman-1638162/)

## Multi-view

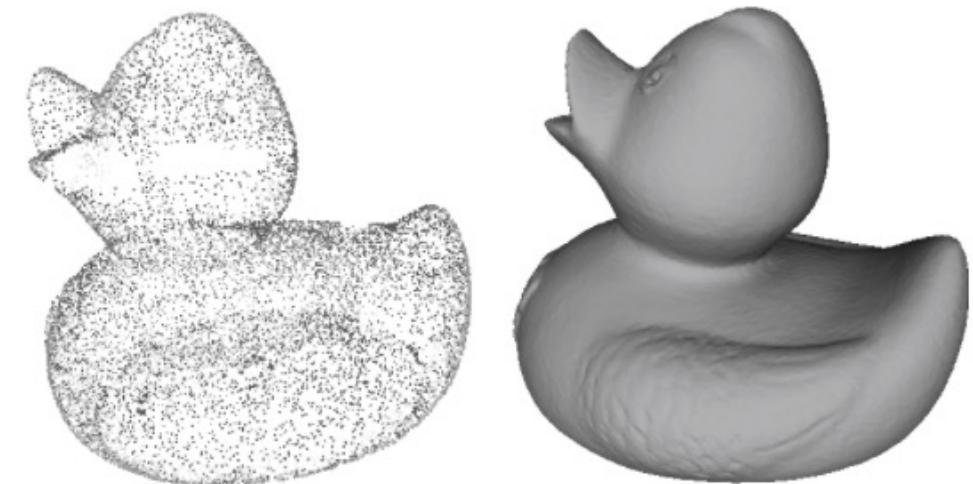
- ✓ 2D CNN
- Generalize to points?
- ✗ Large networks



[http://photoboothexpo.com/  
bullet-time-photo-booths/](http://photoboothexpo.com/bullet-time-photo-booths/)

## Point Cloud/Mesh

- ✓ Raw format/Efficiency
- Explicit representation
- ✗ Unorganized/Unordered

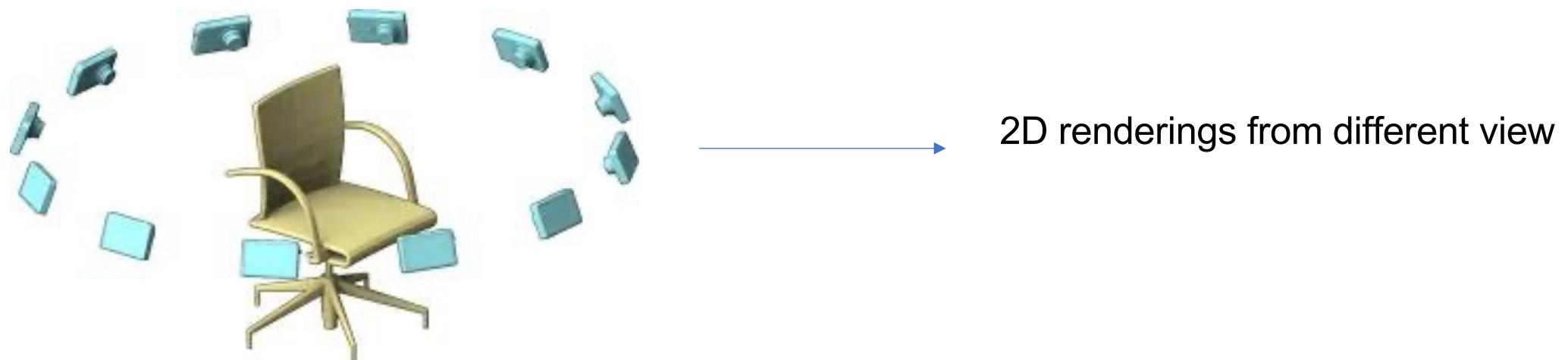


[https://elmoatazbill.users.greyc.fr/  
point\\_cloud/reconstruction.png](https://elmoatazbill.users.greyc.fr/point_cloud/reconstruction.png)



# Multi-view Convolutional Neural Networks for 3D Shape Recognition

- This paper presents a novel CNN architecture that combines information from multiple views of a 3D shape into a single and compact shape descriptor offering even better recognition performance.
- The network operates on 2D renderings of a 3D object and still performs better than those that directly build on the 3D representations





# Multi-view Convolutional Neural Networks – Why ?

- One main reason for working with images is the relative efficiency of the 2D versus the 3D representations.
- For example, 3D ShapeNets use a coarse representation of shape, a **30×30×30 grid of binary voxels**. In contrast a single projection of the 3D model of the same input size corresponds to an image of **164×164 pixels**.
- Another advantage of using 2D representations is that we can leverage
  - (i) Advances in image descriptors
  - (ii) massive image databases (ImageNet) to pre-train our CNN architectures



# Multi-view Convolutional Neural Networks - Approach

- Approach is to learn to combine information from multiple views using a unified CNN architecture that includes a view-pooling layer. All the parameters of the CNN architecture are learned discriminatively to produce a single compact descriptor for the 3D shape.

Capture images of the 3D shape from different views

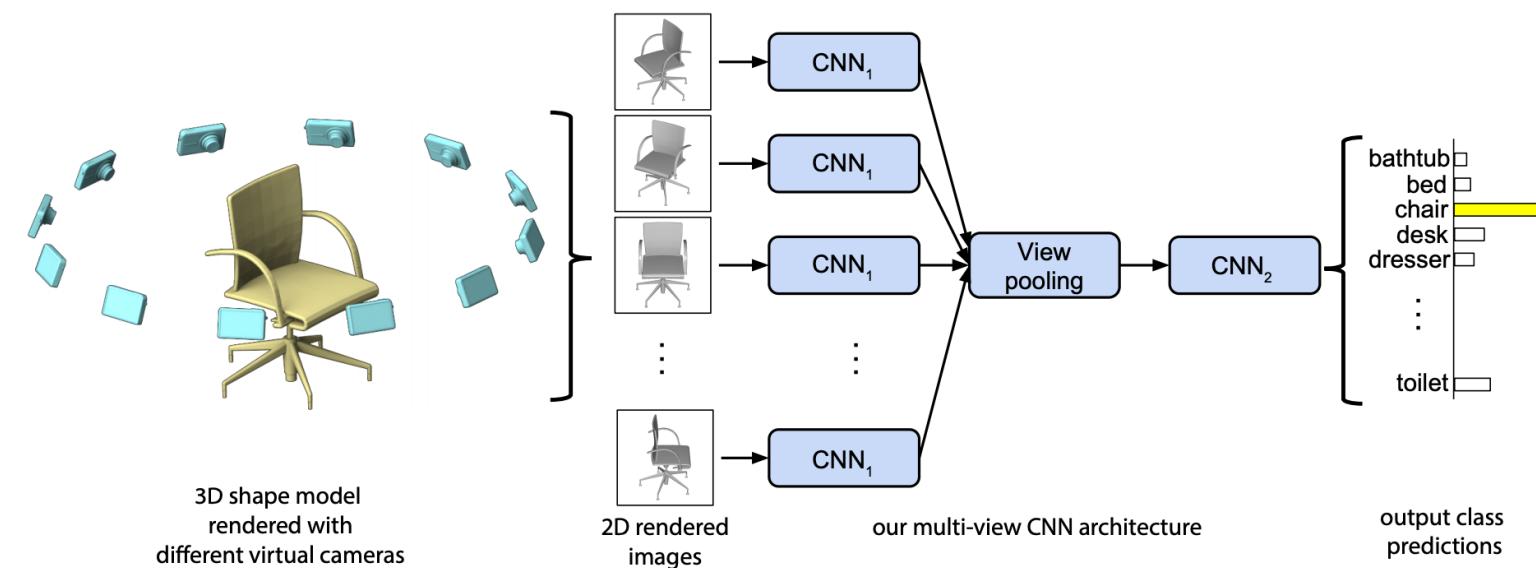


Figure 1. Multi-view CNN for 3D shape recognition (illustrated using the 1<sup>st</sup> camera setup). At test time a 3D shape is rendered from 12 different views and are passed thorough CNN<sub>1</sub> to extract view based features. These are then pooled across views and passed through CNN<sub>2</sub> to obtain a compact shape descriptor.



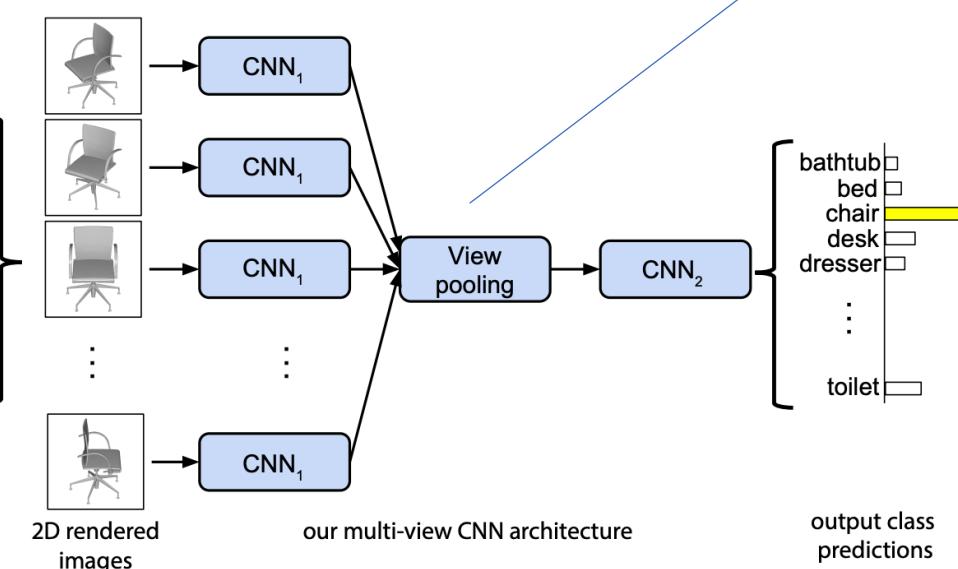
# Multi-view Convolutional Neural Networks - Approach

Capture images of the 3D shape from different views



3D shape model  
rendered with  
different virtual cameras

Pass the rendered images through common CNN



Element wise pooling across feature maps corresponding to all renderings

Output descriptor is used to classify the shape using SVM

Figure 1. Multi-view CNN for 3D shape recognition (illustrated using the 1<sup>st</sup> camera setup). At test time a 3D shape is rendered from 12 different views and are passed thorough CNN<sub>1</sub> to extract view based features. These are then pooled across views and passed through CNN<sub>2</sub> to obtain a compact shape descriptor.



# Multi-view Convolutional Neural Networks - Results

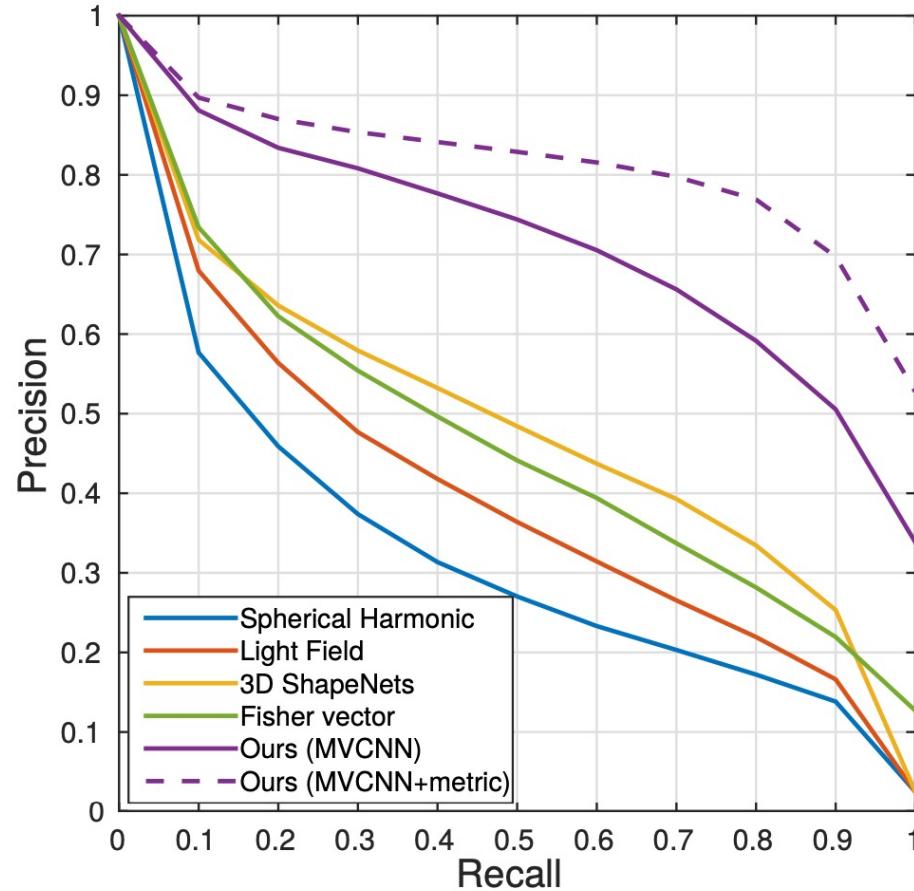


Figure 2. Precision-recall curves for various methods for 3D shape retrieval on the ModelNet40 dataset. Our method significantly outperforms the state-of-the-art on this task achieving 80.2% mAP.

Method	Aug.	Accuracy
(1) FV [30]	-	79.0%
(2) CNN M	-	77.3%
(3) CNN M, fine-tuned	-	84.0%
(4) CNN M, fine-tuned	6×	85.5%
(5) MVCNN M, fine-tuned	6×	86.3%
(6) CNN VD	-	69.3%
(7) CNN VD, fine-tuned	-	86.3%
(8) CNN VD, fine-tuned	6×	86.0%
(9) MVCNN VD, fine-tuned	6×	<b>87.2%</b>
(10) Human performance	n/a	93.0%

Table 2. Classification results on SketchClean. Fine-tuned CNN models significantly outperform Fisher vectors [30] by a significant margin. MVCNNs are better than CNN trained with data jittering. The results are shown with two different CNN architectures – VGG-M (row 2-5) and VGG-VD (row 6-9).



# Multi-view Convolutional Neural Networks – Gradient Visualization

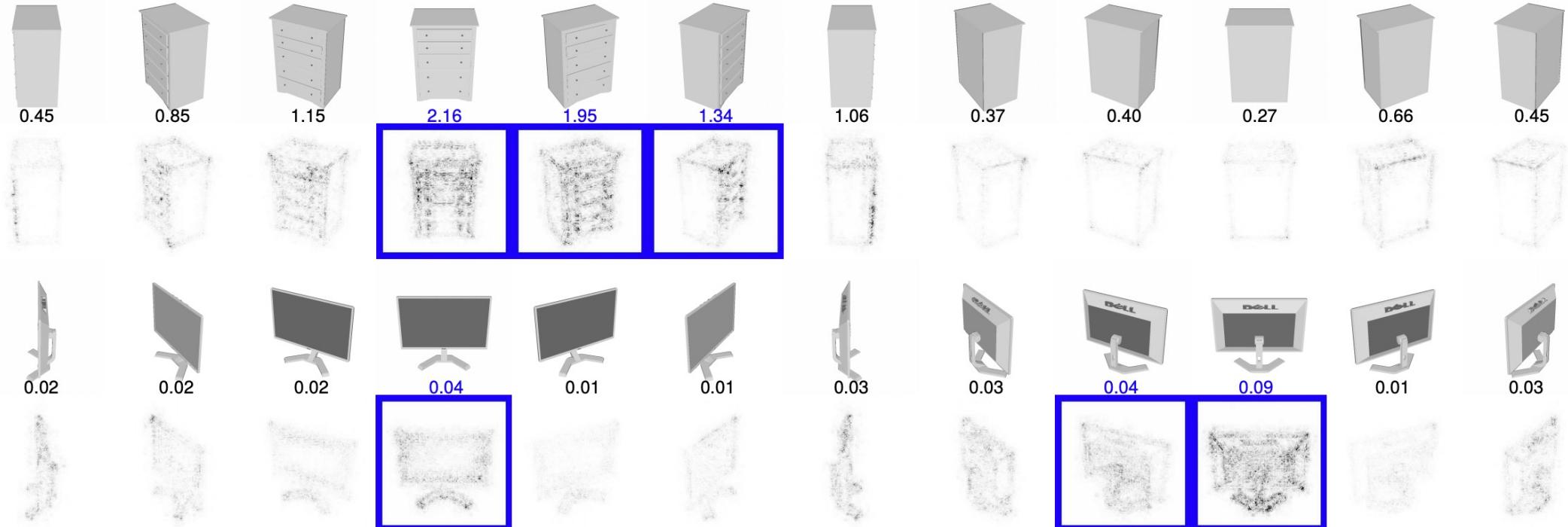


Figure 3. Top three views with the highest saliency are highlighted in blue and the relative magnitude of gradient energy for each view is shown on top. The saliency maps are computed by back-propagating the gradients of the class score onto the images via the view-pooling layer. Notice that the handles of the dresser are the most discriminative features. (Figures are enhanced for visibility.)



## VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition

- Range sensors such as LiDAR and RGBD cameras are increasingly found in modern robotic systems, providing a rich source of 3D information that can aid in this task.
- However, many current systems do not fully utilize this information and have trouble efficiently dealing with large amounts of point cloud data.
- In this paper, VoxNet is introduced as an architecture to tackle this problem by integrating a Volumetric Occupancy Grid representation with a supervised 3D Convolutional Neural Network (3D CNN).



# Input Representation : Volumetric Occupancy Grid

- Occupancy grids represent the state of the environment as a **3D lattice of random variables** (each corresponding to a voxel) and maintain a probabilistic estimate of their occupancy as a function of incoming sensor data and prior knowledge.
- Reason for Using Occupancy Grid as Input Representation
  - (i) They allow us to **efficiently estimate** free, occupied and unknown space from range measurements.
  - (ii) They can be stored and manipulated with simple and **efficient data structures**.

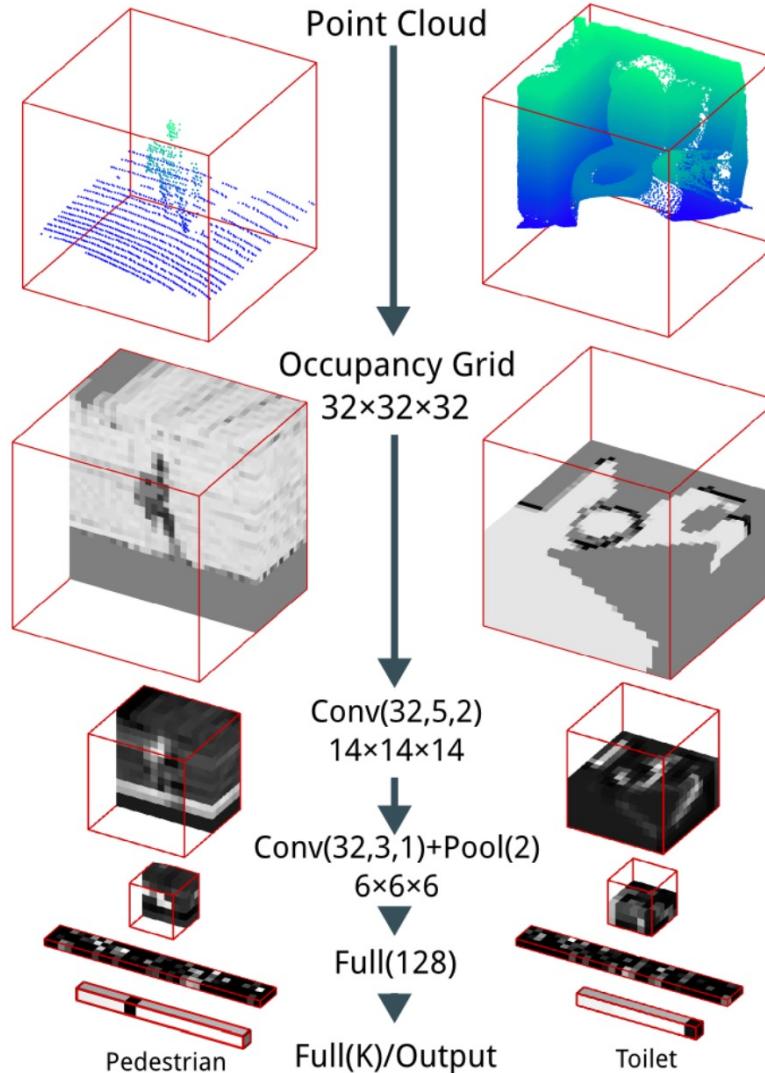


# 3D Convolutional Network Layers

- Input Layer: This layer accepts a fixed-size grid of  $I \times J \times K$  voxels. In this work, we use  $I = J = K = 32$
- Convolutional Layers: These layers accept 4 dimensional input volumes in which three of the dimensions are spatial, and the fourth contains the feature maps. The layer creates  $f$  feature maps by convolving the input with  $f$  learned filters of shape  $d \times d \times d \times f'$ , where  $d$  are the spatial dimensions and  $f'$  is the number of input feature maps.
- Pooling Layers: These layers downsample the input volume by a factor of  $m$  along the spatial dimensions
- Fully Connected Layer: Fully connected layers have  $n$  output neurons. The output of each neuron is a learned linear combination of all the outputs from the previous layer, passed through a nonlinearity.



# Network Architecture



Probabilistic estimate  
of occupancy

Sequence of range measurements that either hit or passes  
through a given voxel with coordinates  $(i, j, k)$

$$l_{ijk}^t = l_{ijk}^{t-1} + z^t l_{\text{occ}} + (1 - z^t) l_{\text{free}} \quad (1)$$

where  $l_{\text{occ}}$  and  $l_{\text{free}}$  are the log odds of the cell being occupied  
or free given that the measurement hit or missed the cell,

Filters at the input level encode spatial structures  
such as planes and corners at different  
orientations.

The output of the network is probabilities for  
each class. **Loss is Multinomial negative log-  
likelihood and Evaluation metric is Accuracy**



# Visualizations and Results

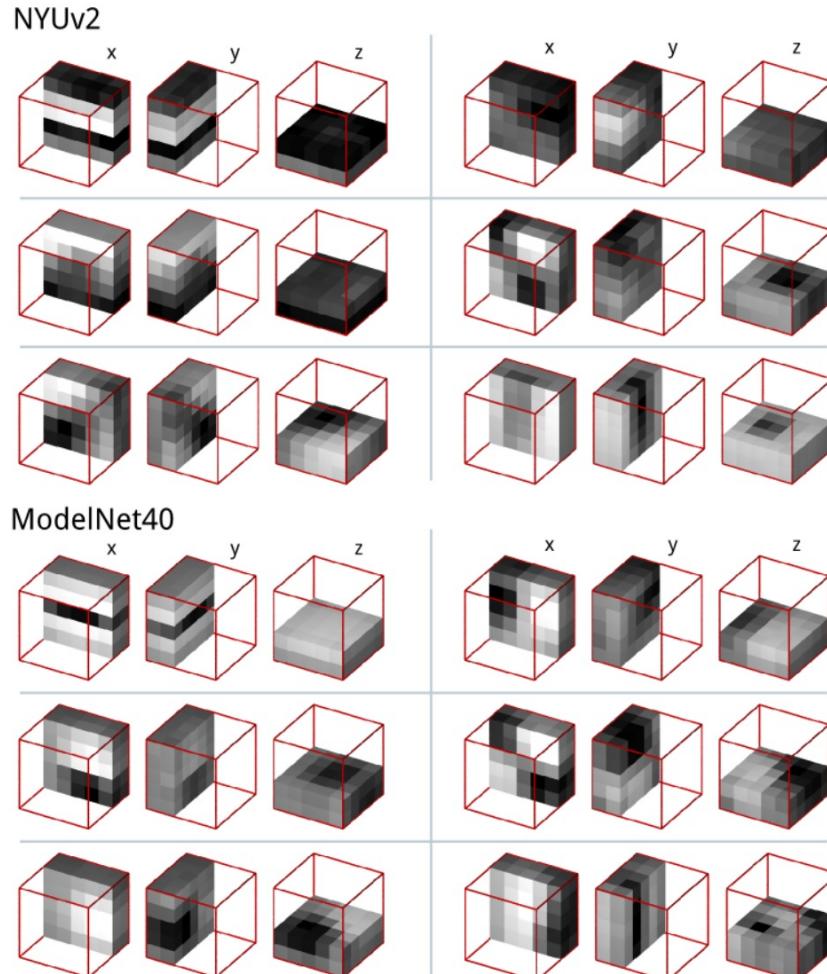


Fig. 4. Cross sections along the  $x$ ,  $y$  and  $z$  axes of selected first layer filters learned in the ModelNet40 and NYUv2 datasets.

COMPARISON WITH OTHER METHODS IN SYDNEY OBJECT DATASET

Method	Avg F1
UFL+SVM[21]	0.67
GFH+SVM[37]	0.71
Multi Resolution VoxNet	<b>0.73</b>

TABLE IV  
COMPARISONS WITH SHAPENET IN MODELNET (AVG ACC)

Dataset	ShapeNet	VoxNet
ModelNet10	0.84	<b>0.92</b>
ModelNet40	0.77	<b>0.83</b>

TABLE V  
COMPARISON WITH SHAPENET IN NYUV2 (AVG ACC)

Dataset	ShapeNet	VoxNet	VoxNet Hit
NYU	0.58	<b>0.71</b>	0.70
ModelNet10→NYU	<b>0.44</b>	0.34	0.25



# PointNet - What is Point Cloud?

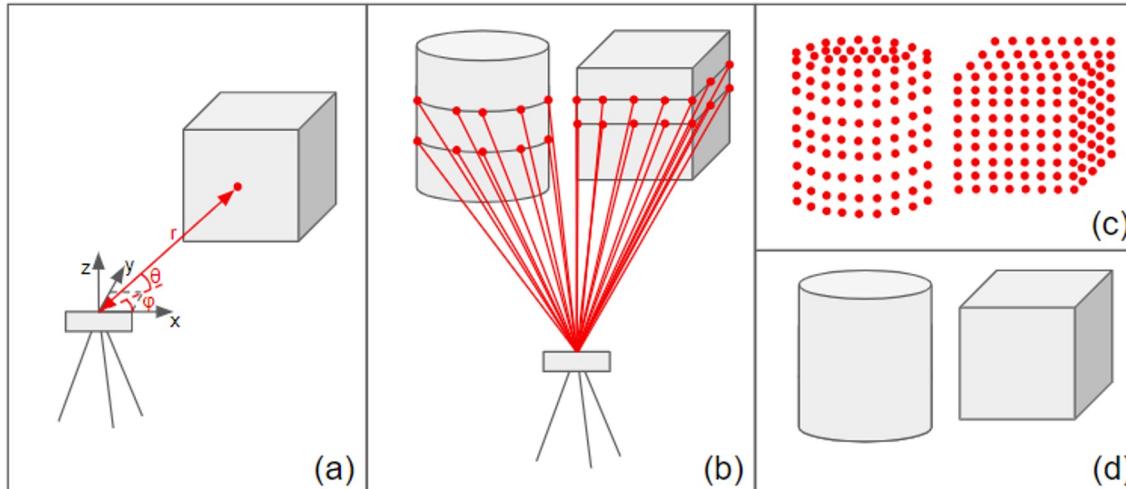


Figure 1: A point cloud from a laser scanner. See body text for more explanation.



A Trimble 3D scanning unit sweeps an area to obtain point cloud data.

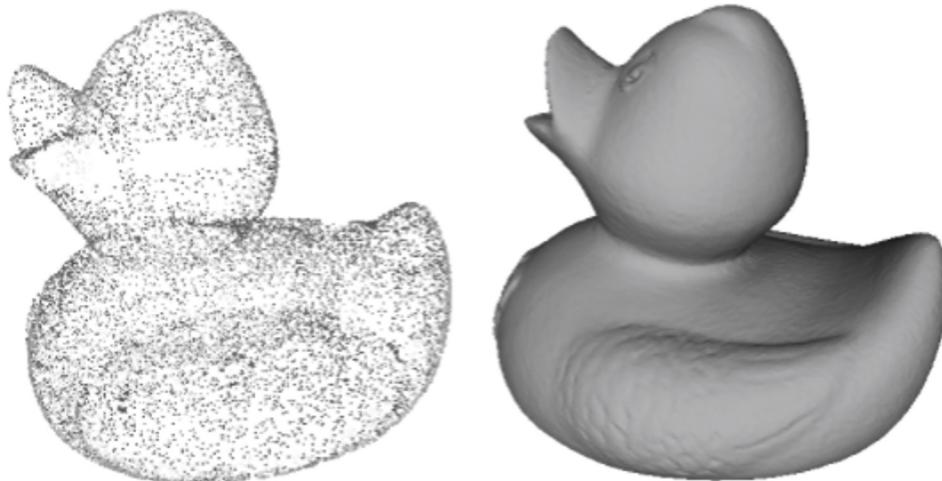
- A point cloud is nothing more than a collection of millions (sometimes billions) of points coming from a scanner. (Unorganized)
- The points only represent the surfaces of scanned objects
- Typical Scanners: 2D or 3D Lidar (Laser Scanner), Structure Sensor (1st Gen Kinect)



# PointNet - What is Point Cloud?

## Point Cloud/Mesh

- Raw format/Efficiency
- Explicit representation
  - ✗ Unorganized/Unordered



```
ply
format ascii 1.0
element vertex 75463
property float x
property float y
property float z
property float nx
property float ny
property float nz
end_header
-472.25 -377.93 579.102 922.061 74.0133 -379.902
-471.68 -377.93 580.412 906.271 50.2959 -419.695
-468.945 -383.789 584.961 918.904 65.9779 -388.924
-466.14 -377.93 590.82 878.145 37.8206 -476.897
-469.165 -377.93 584.961 881.953 13.8531 -471.134
```

### Point cloud format example:

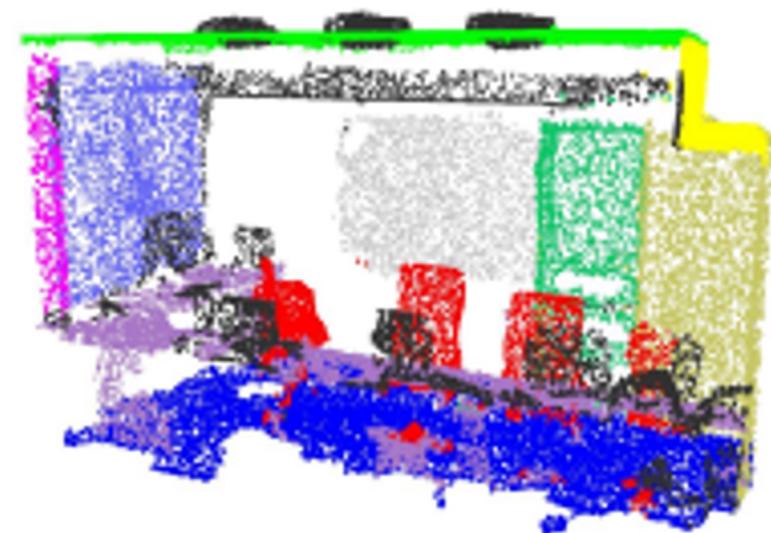
- coordinate of each point ( $x, y, z$ )
- normal coordinate of each point ( $nx, ny, nz$ )  
(if have)



# Machine Learning Tasks on Point Cloud Data

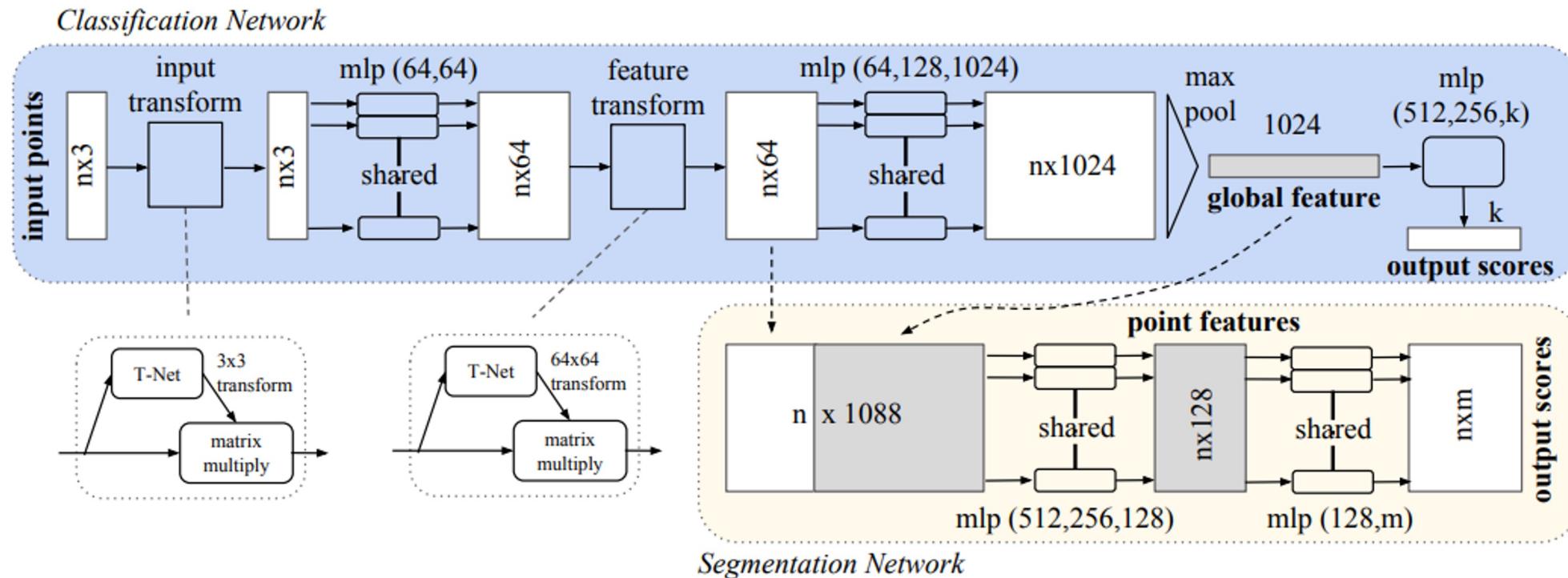


Classification





# PointNet: A Framework to Process Point Cloud with MLP



**Figure 2. PointNet Architecture.** The classification network takes  $n$  points as input, applies input and feature transformations, and then aggregates point features by max pooling. The output is classification scores for  $k$  classes. The segmentation network is an extension to the classification net. It concatenates global and local features and outputs per point scores. “mlp” stands for multi-layer perceptron, numbers in bracket are layer sizes. Batchnorm is used for all layers with ReLU. Dropout layers are used for the last mlp in classification net.



# Results of PointNet

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

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

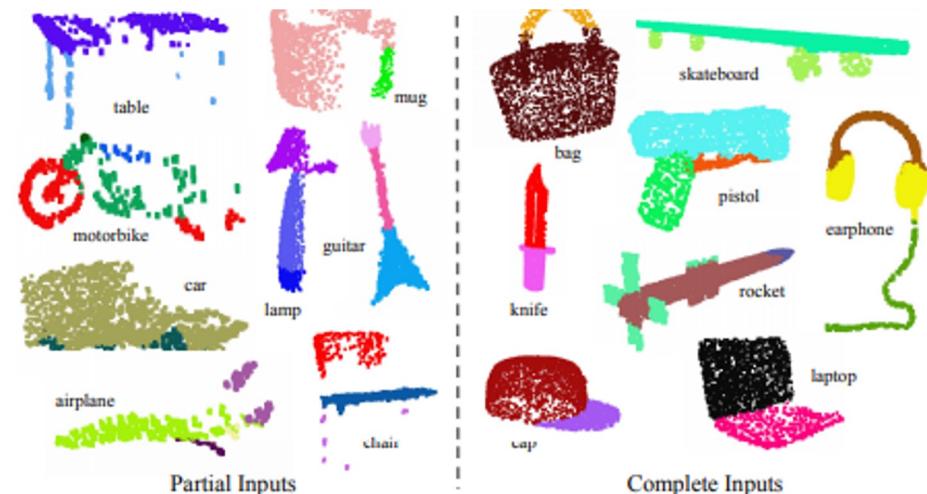


Figure 3. Qualitative results for part segmentation. We visualize the CAD part segmentation results across all 16 object categories. We show both results for partial simulated Kinect scans (left block) and complete ShapeNet CAD models (right block).



# Results of PointNet

	mean	aero	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	skate board	table
# shapes		2690	76	55	898	3758	69	787	392	1547	451	202	184	283	66	152	5271
Wu [27]	-	63.2	-	-	-	73.5	-	-	-	74.4	-	-	-	-	-	-	74.8
Yi [29]	81.4	81.0	78.4	77.7	<b>75.7</b>	87.6	61.9	<b>92.0</b>	85.4	<b>82.5</b>	<b>95.7</b>	<b>70.6</b>	91.9	<b>85.9</b>	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	<b>83.7</b>	<b>83.4</b>	<b>78.7</b>	<b>82.5</b>	74.9	<b>89.6</b>	<b>73.0</b>	91.5	<b>85.9</b>	80.8	95.3	65.2	<b>93.0</b>	81.2	<b>57.9</b>	<b>72.8</b>	<b>80.6</b>

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



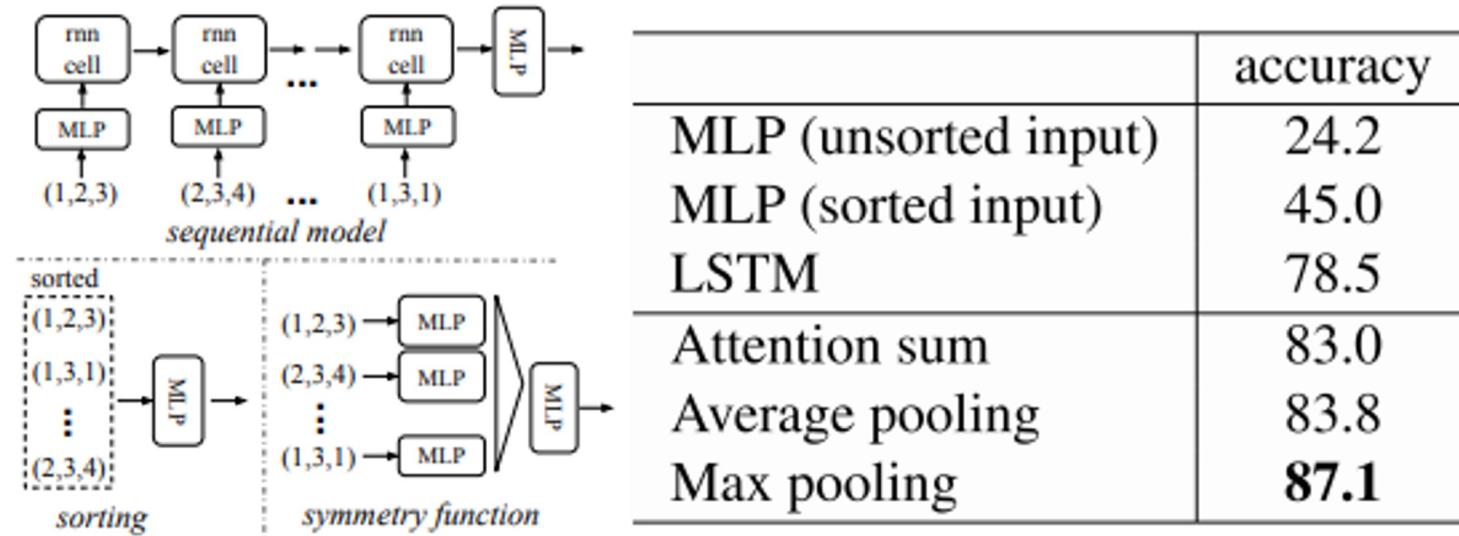
Figure 4. **Qualitative results for semantic segmentation.** Top row is input point cloud with color. Bottom row is output semantic segmentation result (on points) displayed in the same camera viewpoint as input.

	mean IoU	overall accuracy
Ours baseline	20.12	53.19
Ours PointNet	<b>47.71</b>	<b>78.62</b>

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



# How Does PointNet Achieve Point Order Invariance?



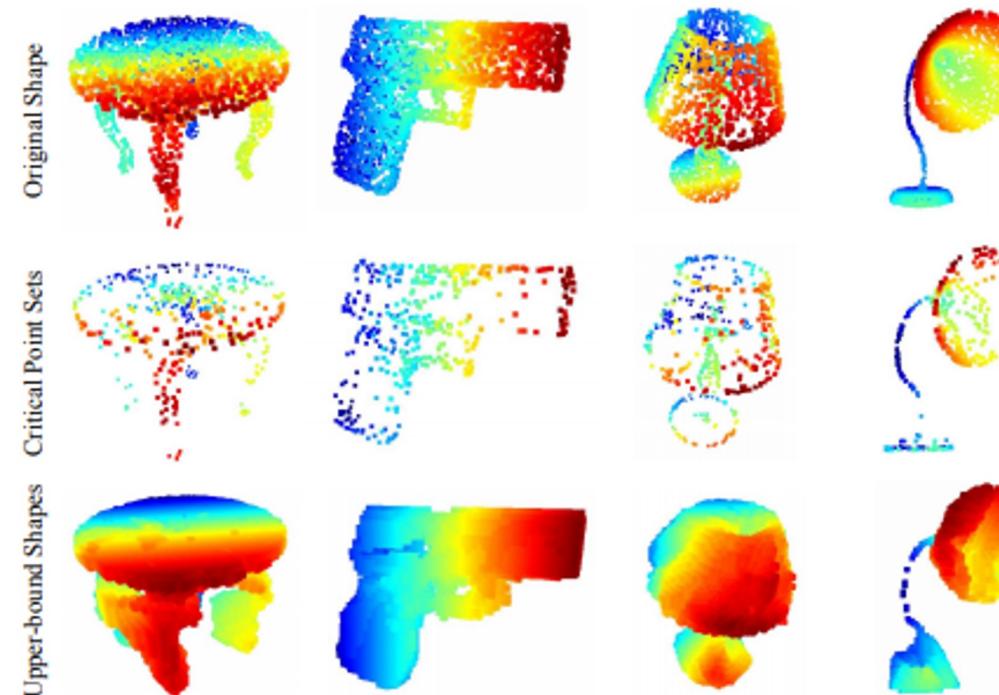
**Figure 5. Three approaches to achieve order invariance.** Multi-layer perceptron (MLP) applied on points consists of 5 hidden layers with neuron sizes 64,64,64,128,1024, all points share a single copy of MLP. The MLP close to the output consists of two layers with sizes 512,256.



# Visualizing PointNet

Critical point sets: those contributed to the max pooled feature, summarizing the skeleton of the shape.

Upper-bound shapes: the largest possible point cloud that give the same global feature shape.



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



# PointNet++

PointNet++ leverages neighborhoods at multiple scales to achieve both robustness and detail capture.

Introduces a hierarchical neural network that applies PointNet recursively on a nested partitioning of the input point set.

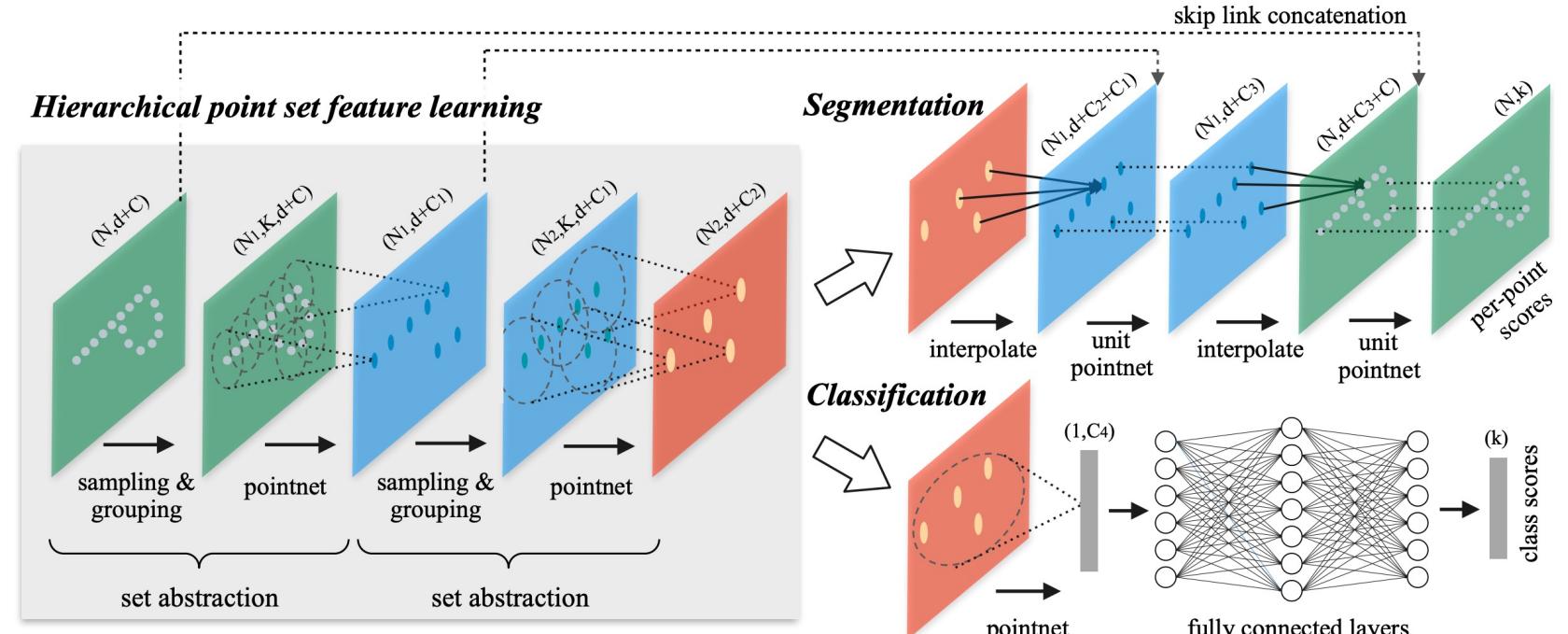
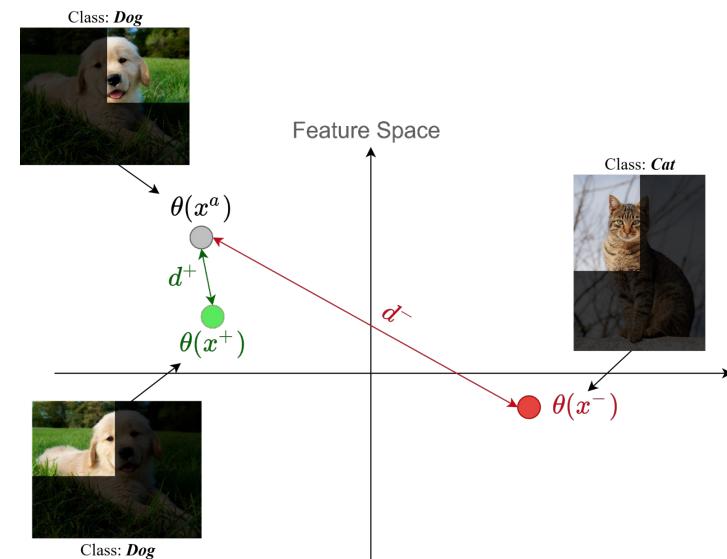
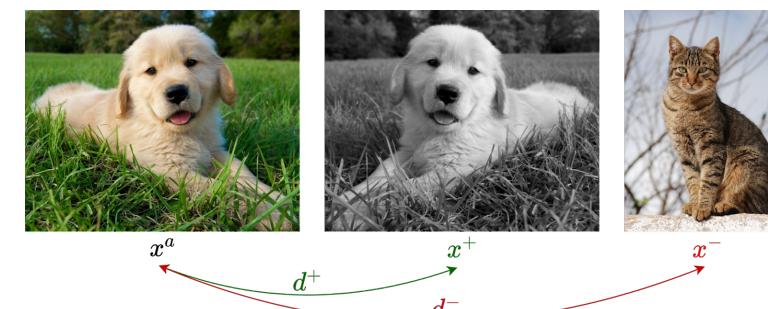
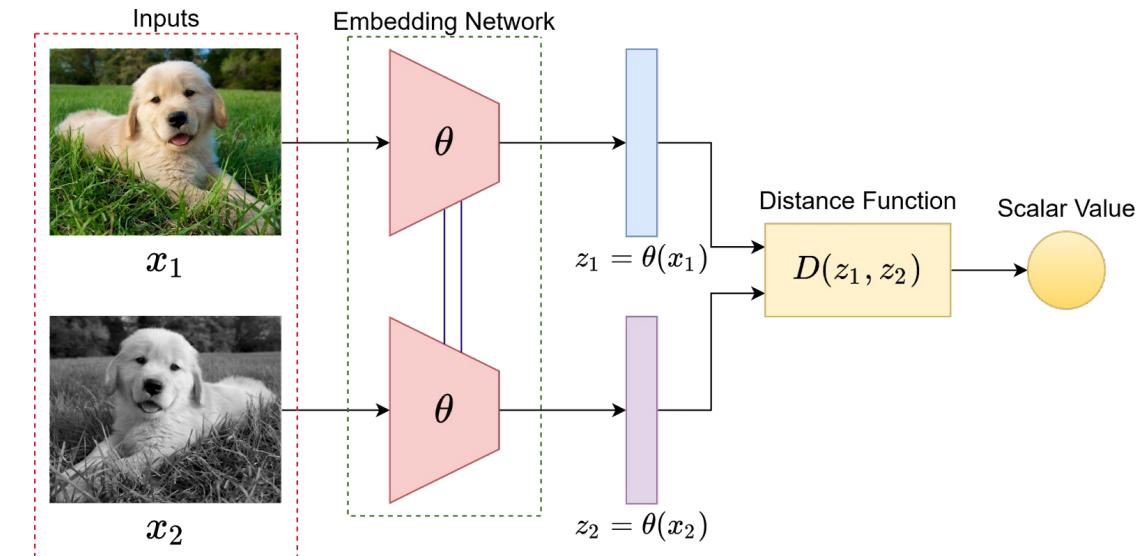


Figure 2: Illustration of our hierarchical feature learning architecture and its application for set segmentation and classification using points in 2D Euclidean space as an example. Single scale point grouping is visualized here. For details on density adaptive grouping, see Fig. 3



# How to Learn Representations without (**External**) Supervision

- Self-supervised learning (SSL)
- Benefit: scalability
- Two (very coarse) categorization
  - Contrastive
    - MoCo
    - SimCLR
    - SwAV
  - **Reconstructive / Generative**
    - VAE
    - Mask Autoencoder (MAE)
    - Cycle-consistency





# Richard Feynman Quote

What I cannot create,  
I do not understand.

I know how to solve every  
problem that has been solved

Why const  $\times$  soft PC

To LEARN:

Bethe Ansatz Prob.  
Kondo  
2-D Hall  
accel. Temp  
Non linear Classical Hydro

(A)  $f = u(r, \alpha)$

$g = 4(r-z)u(r,z)$

(B)  $f = 1/r \cdot \alpha / (u \cdot \alpha)$

© Copyright California Institute of Technology. All rights reserved.  
Commercial use or modification of this material is prohibited.

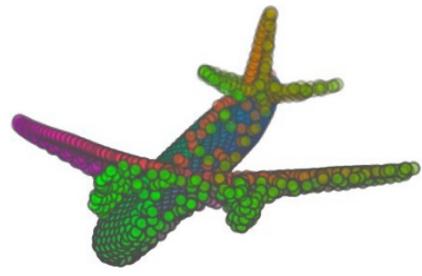
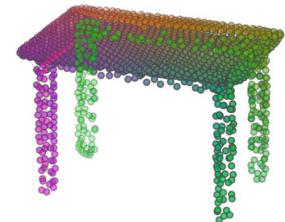
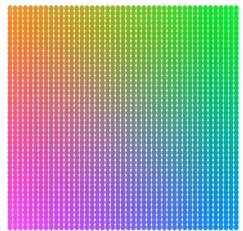
*"What I cannot create, I do not understand."*



Richard Feynman (1918-1988)  
Theoretical Physicist  
Won 1965 Nobel Prize in Physics



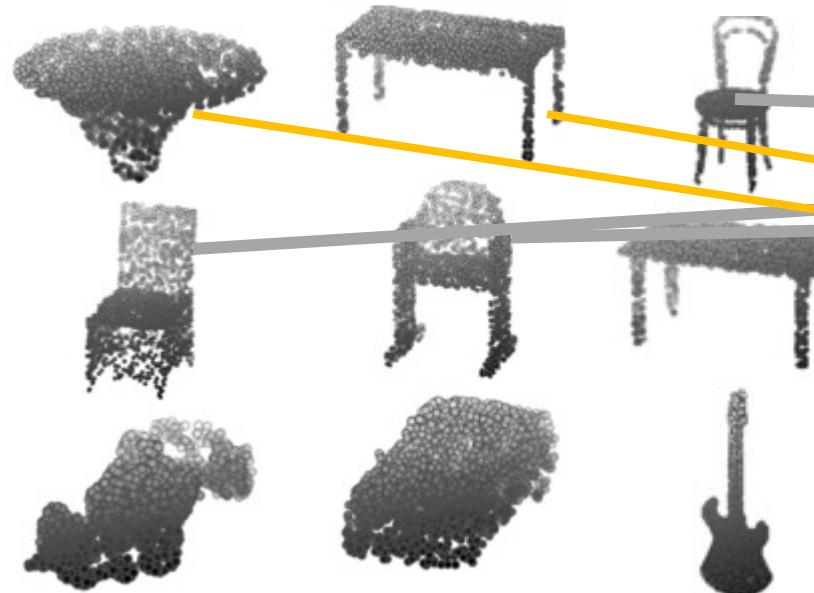
# Can Neural Networks Learn Paper Folding?



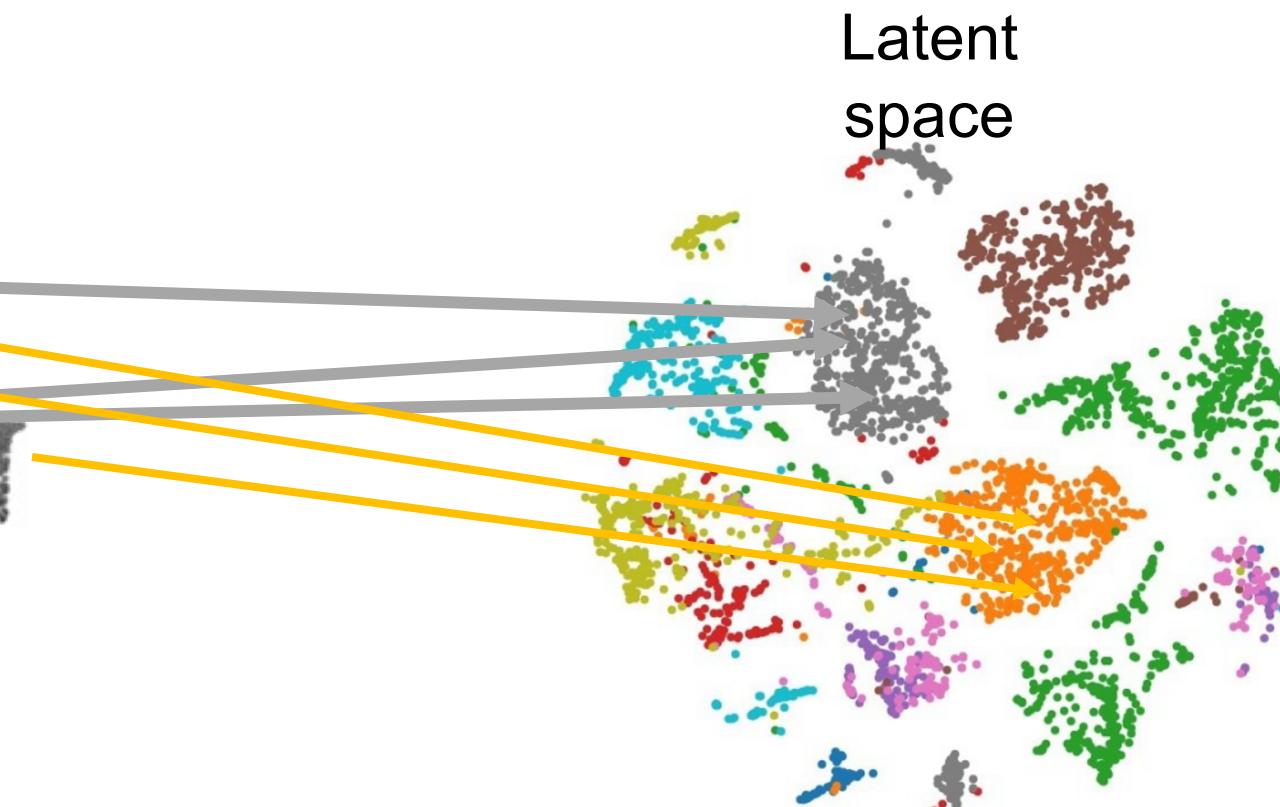


# Representation Learning for Point Clouds

3D Data (Point Clouds)



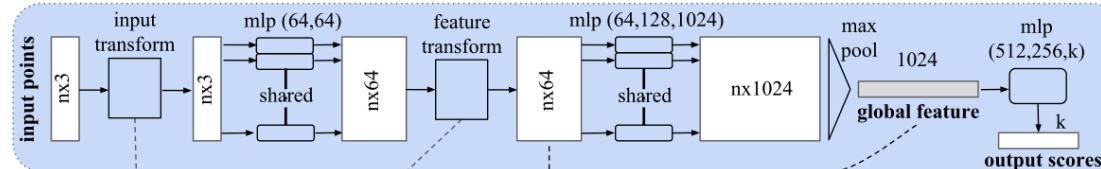
Latent  
space



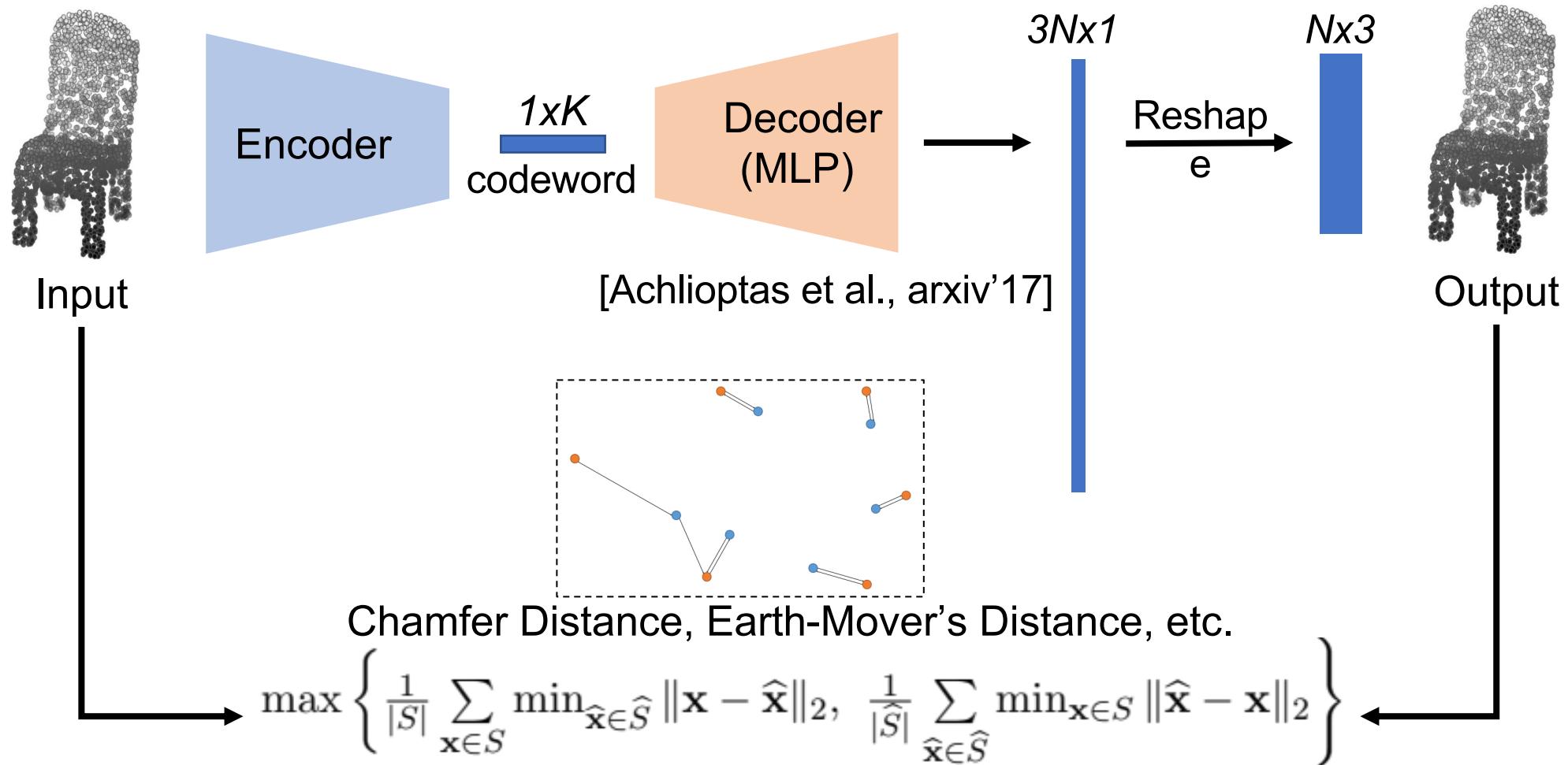
$F(P)$ : point cloud  $\rightarrow$  codeword



# Baseline Point Cloud Auto-encoder



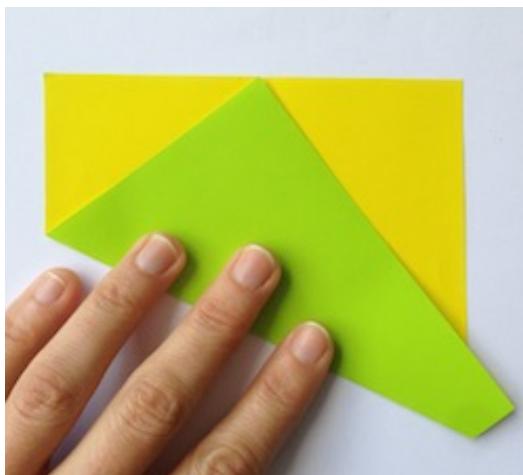
[Qi et al., CVPR'17]



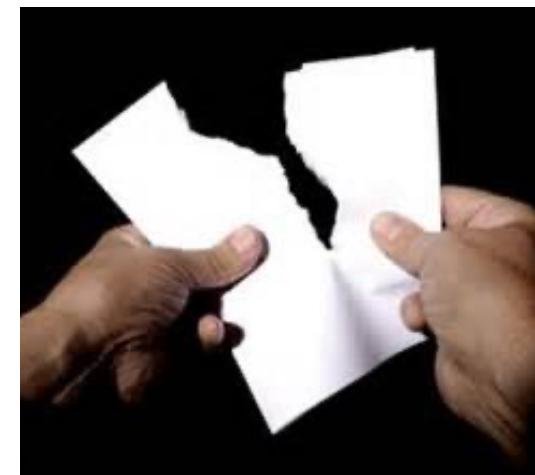


# Intuition of FoldingNet: Elastic Paper Folding

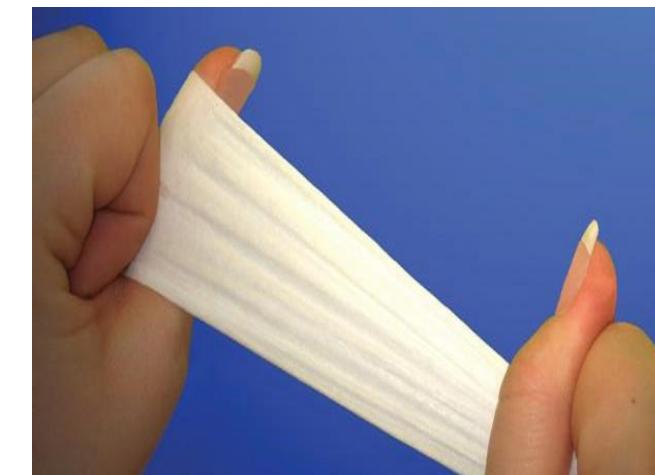
- 3D point clouds are often obtained from object surfaces
  - Discretized from CAD models
  - Sampled from line-of-sight sensors
- 3D object surfaces are intrinsically 2D-manifolds
  - Can be transformed from a 2D plane, through the Origami operations
  - This 2D-3D mapping is known as parameterization/cross-parameterization



Fold



Tear



Stretch



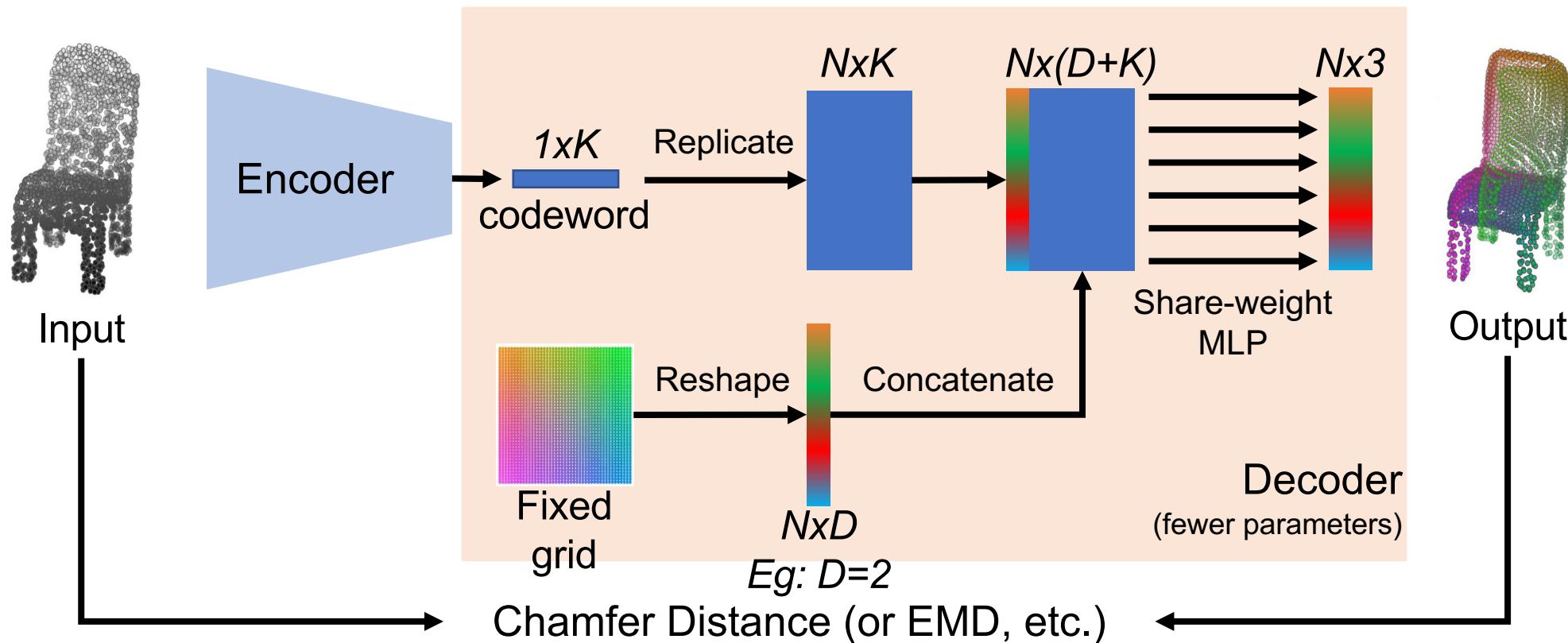
Glue



# FoldingNet Auto-encoder Framework

FoldingNet decoder is a *Deep Parametric Surface*:

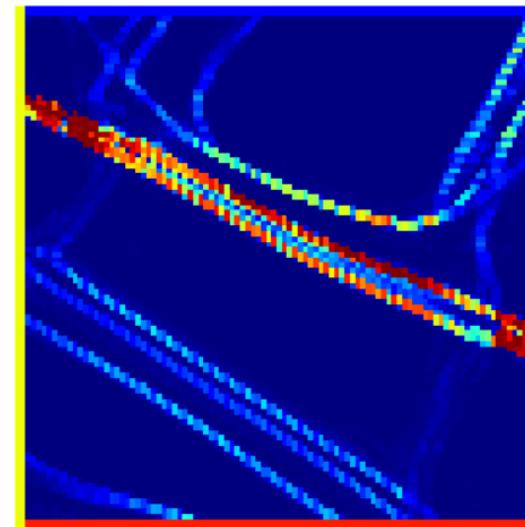
$$x_i = f_{\theta}(g_i, c) \in R^3, \forall g_i \in \text{the fixed/random grid}$$



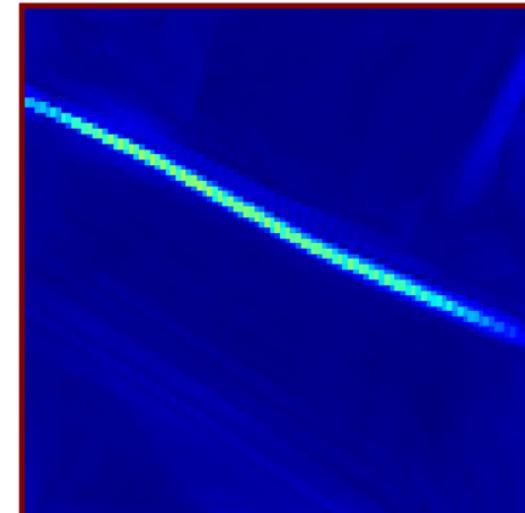


# Learned Folding Profile - Sofa

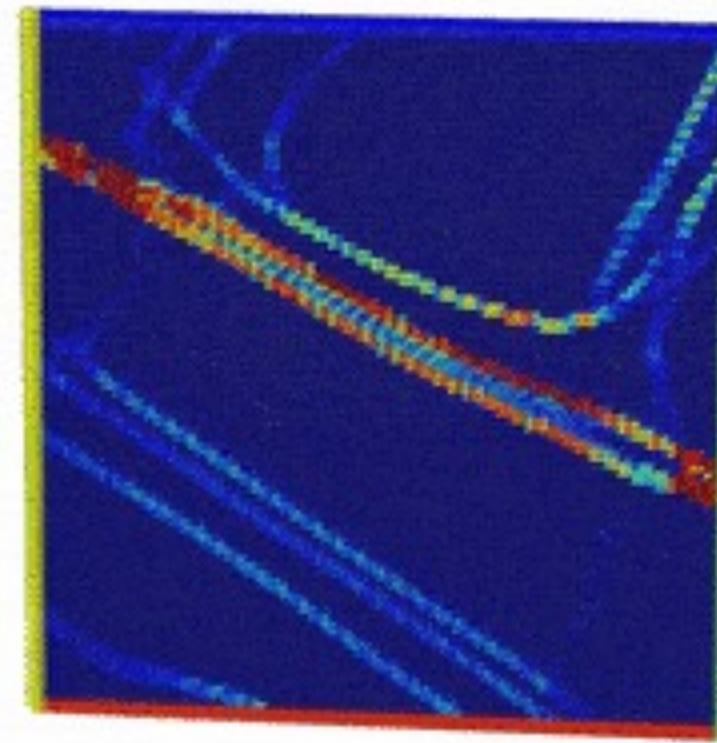
Folding Creases  
(Curvature)



Tear/Stretch  
(Neighbor Distance)



Folding Animation: Sofa  
(colored by curvature)





# Training Process Visualization

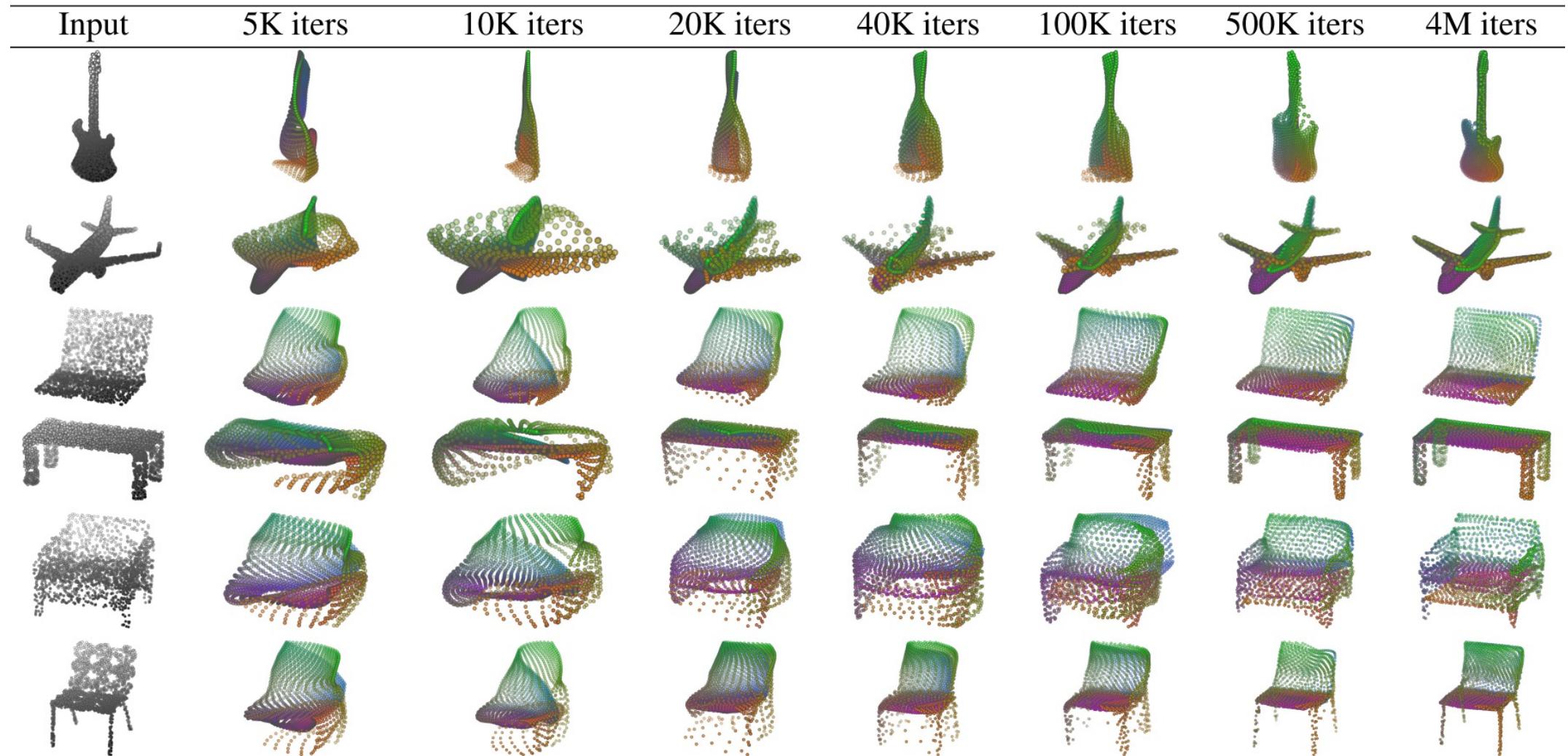


Table 2. Illustration of the training process. Random 2D manifolds gradually transform into the surfaces of point clouds.



# Codeword Space Visualization

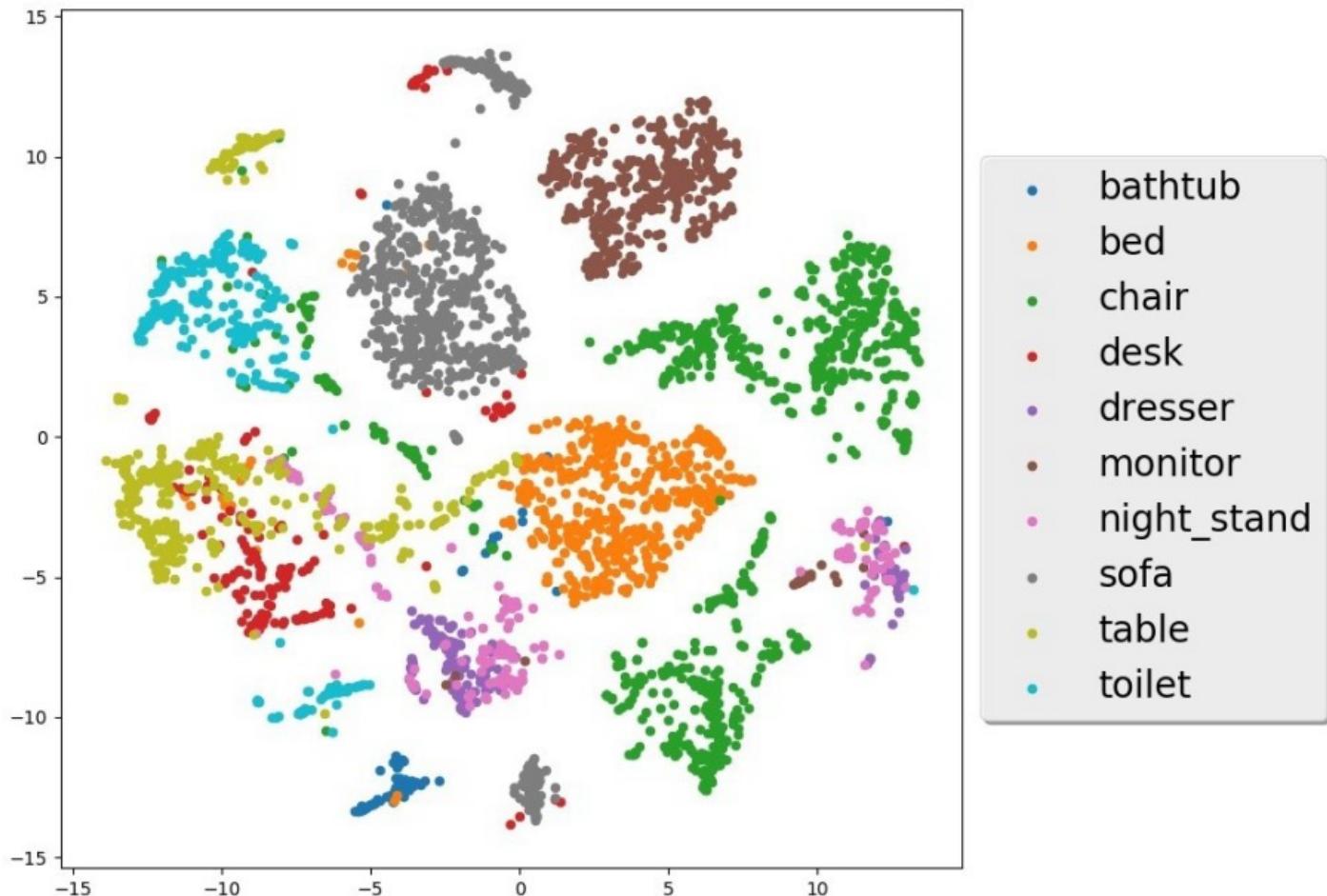


Figure 2. The T-SNE clustering visualization of the codewords obtained from FoldingNet auto-encoder.



# Shape Interpolation

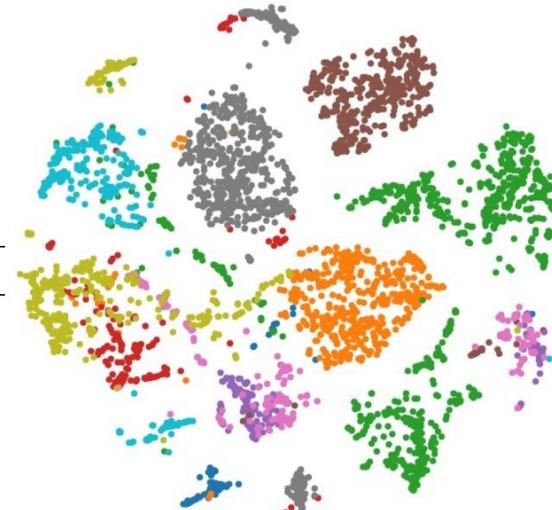
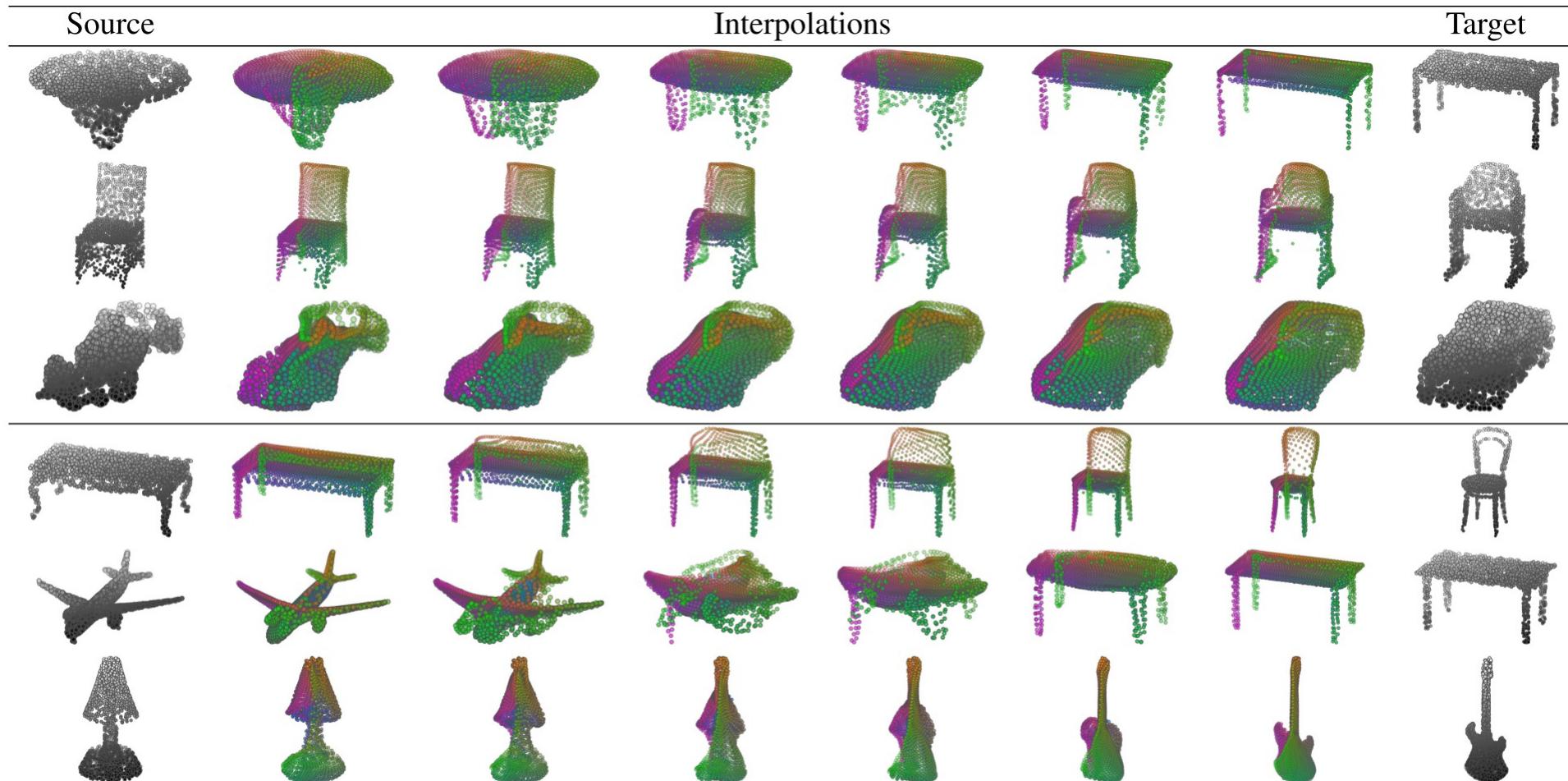


Table 3. Illustration of point cloud interpolation. The first 3 rows: intra-class interpolations. The last 3 rows: inter-class interpolations.



# Quantitative Evaluation of the Learned Features

- Transfer Classification
- Semi-supervised Learning

Method	MN40	MN10
SPH [26]	68.2%	79.8%
LFD [8]	75.5%	79.9%
T-L Network [19]	74.4%	-
VConv-DAE [45]	75.5%	80.5%
3D-GAN [56]	83.3%	91.0%
Latent-GAN [1]	85.7%	<b>95.3%</b>
FoldingNet (ours)	<b>88.4%</b>	94.4%

Table 5. The comparison on classification accuracy between FoldingNet and other unsupervised methods. All the methods train a linear SVM on the high-dimensional representations obtained from unsupervised training.

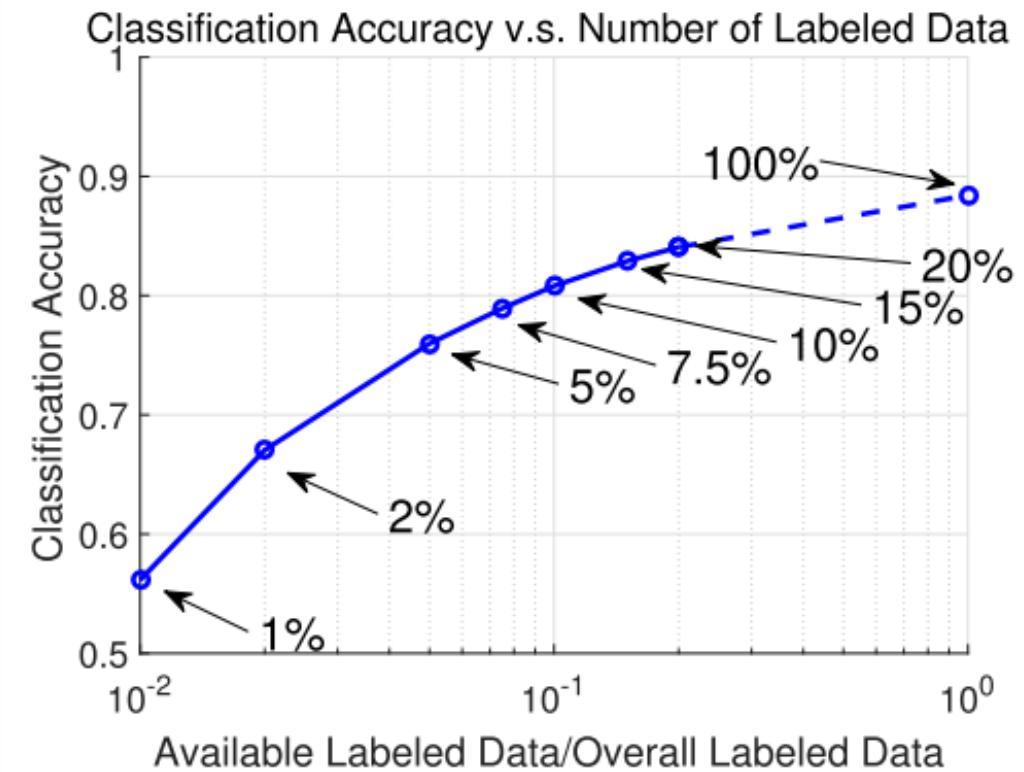


Figure 4. Linear SVM classification accuracy v.s. percentage of available labeled training data in ModelNet40 dataset.



# Ablation: Decoder Variations

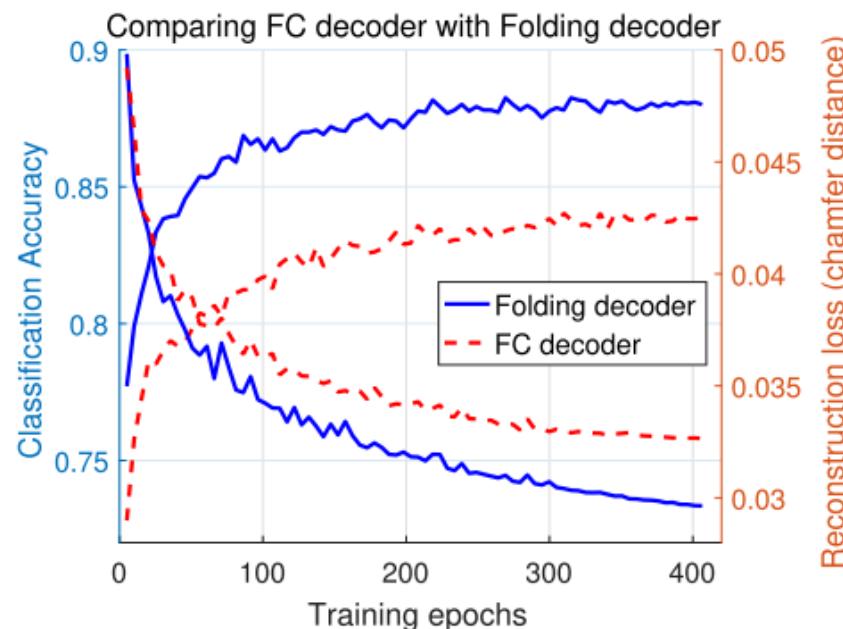


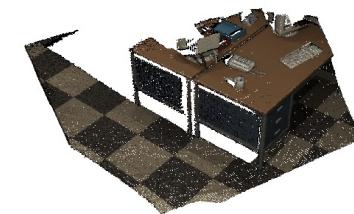
Figure 5. Comparison between the fully-connected (FC) decoder in [1] and the folding decoder on ModelNet40.

Grid Setting	#Folds	Test Cls. Acc.	Test Loss
regular 2D	2	88.25%	0.0296
regular 2D	3	88.41%	0.0290
regular 1D	2	86.71%	0.0355
regular 3D	2	88.41%	0.0284
uniform 2D	2	87.12%	0.0321

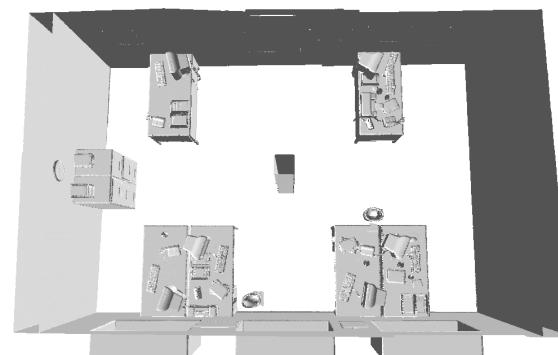
Table 6. Comparison between different FoldingNet decoders. “Uniform”: the grid is uniformly random sampled. “Regular”: the grid is regularly sampled with fixed spacings.

	Cl. Acc.	Tst. Loss	# Params.
FoldingNet	88.41%	0.0296	$1.0 \times 10^6$
Deconv	88.86%	0.0319	$1.7 \times 10^6$

Table 7. Comparison of two different implementations of the folding operation.



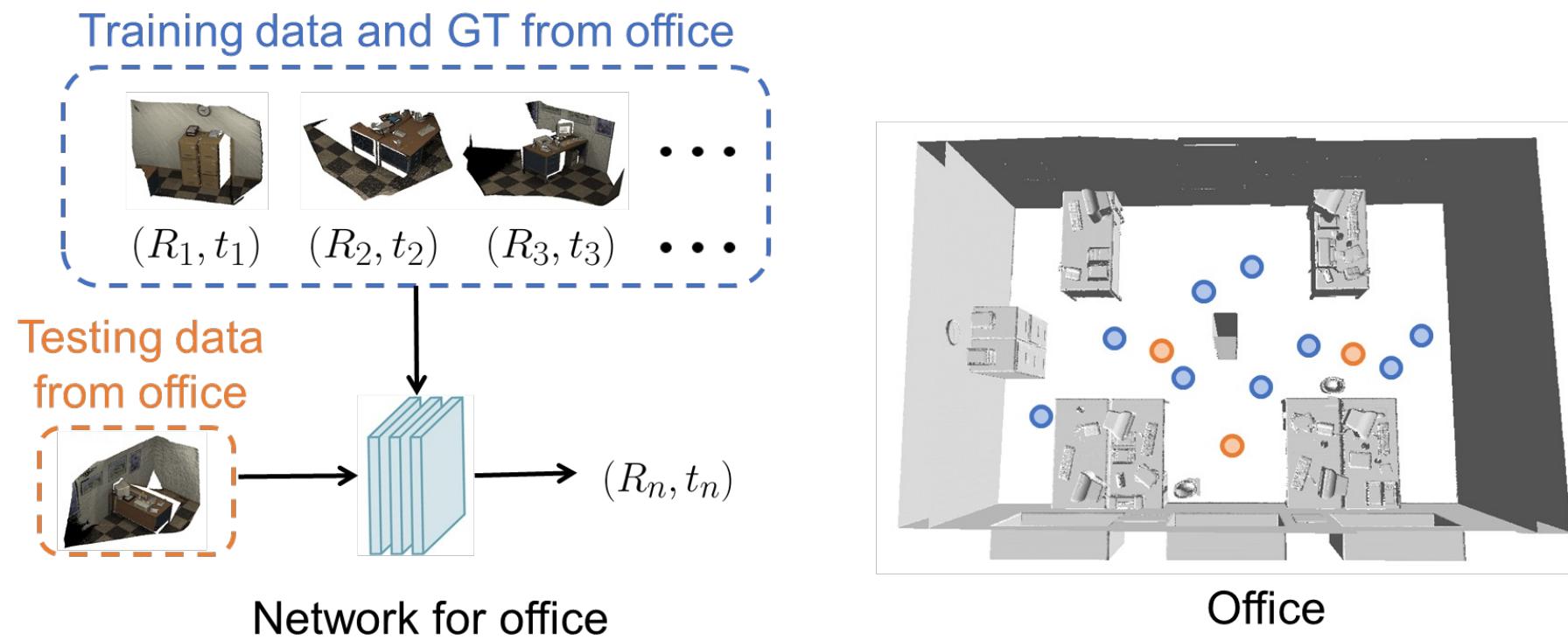
# Can We Do Black-Box Point Cloud Mapping?





# Motivation: Deep Learning for Geometric Vision Problems

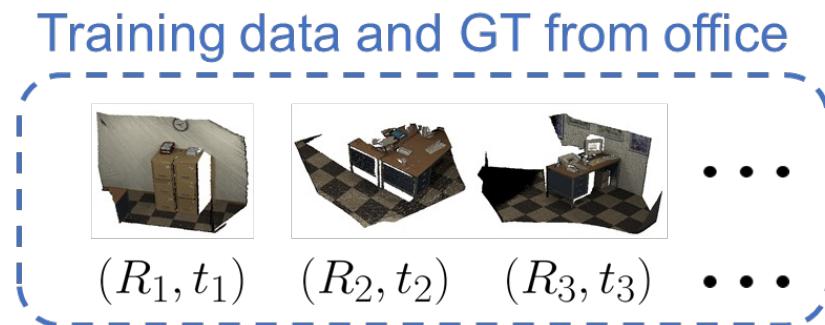
- Common supervised deep learning pipeline for mapping/registration
  - Collect training dataset and ground truth poses
  - Training and testing on the same scene [PoseNet ICCV'15, MapNet CVPR'18, etc.]





# Motivation: Deep Learning for Geometric Vision Problems

- Issues of these supervised approaches
  - Collect training dataset and **ground truth** poses
    - Ground truth is not always easy to obtain: SLAM, surveying, ...

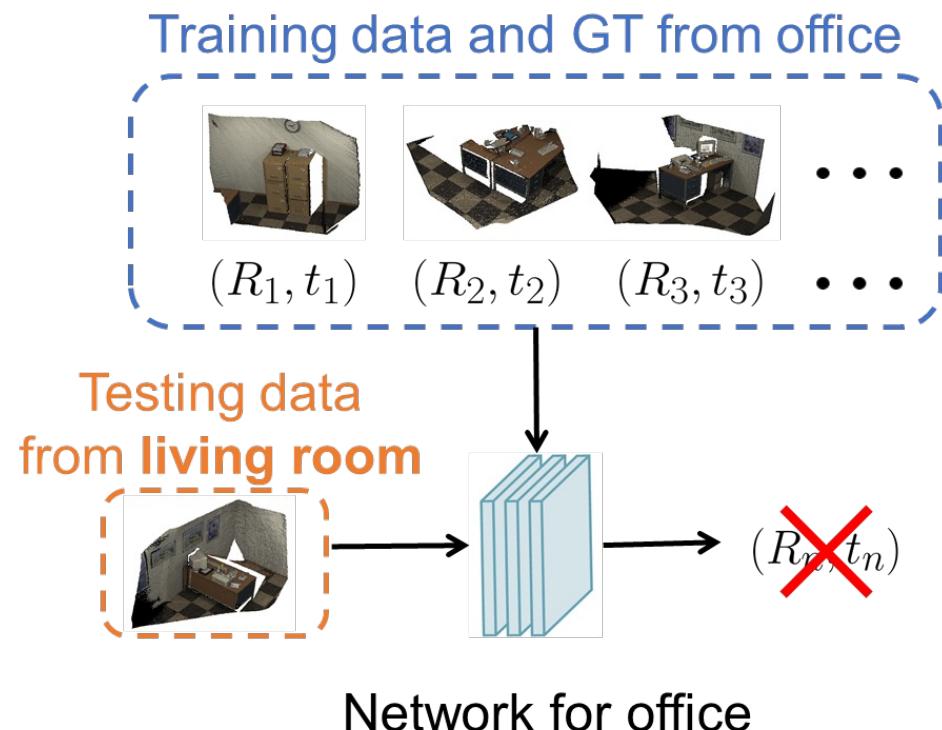


Office

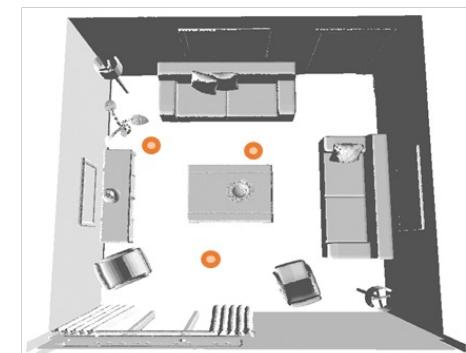


# Motivation: Deep Learning for Geometric Vision Problems

- Issues of these supervised approaches
  - Training and testing on the **same scene**
  - Network trained on “office” might not generalize to “living room”



Office

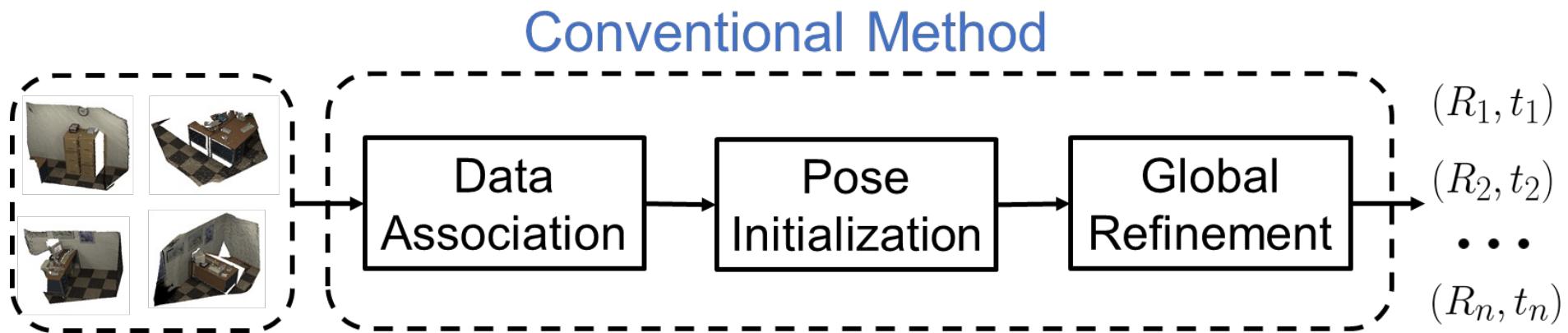


Living room

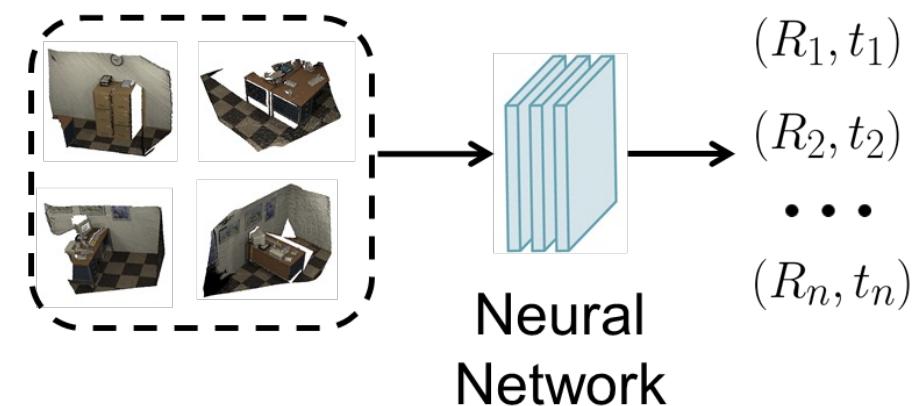


# Our Problem Formulation

- Use neural network to model traditional registration process



## Our Method



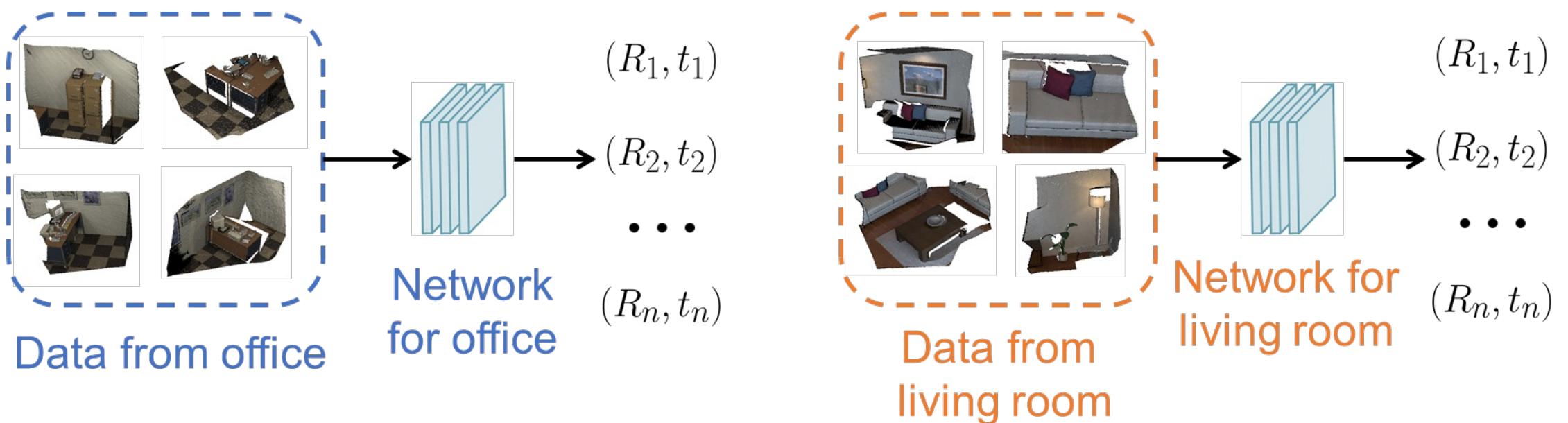
Instead of the common train-and-test pipeline, we propose a new formulation where training the networks is equivalent to solving the mapping problem.

We first use NN to model the complicated registration process that traditionally involves data association, sensor pose initialization, and global refinement.



# Our Problem Formulation

- Use neural network to model traditional registration process
- Train a neural network for each registration problem

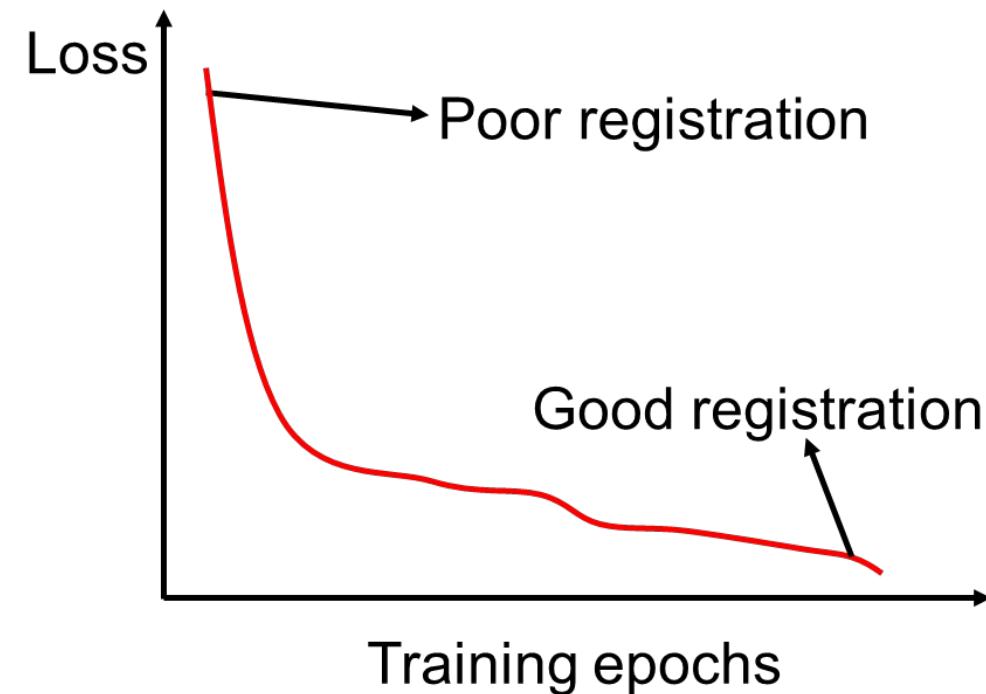
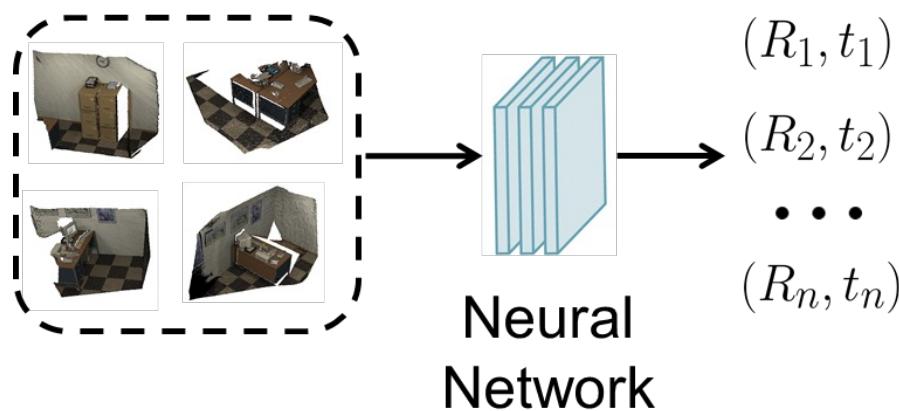


By giving up the expectation of the network should be generalizable, we hope to get something in return: a fully automatic mapping process.



# Our Problem Formulation

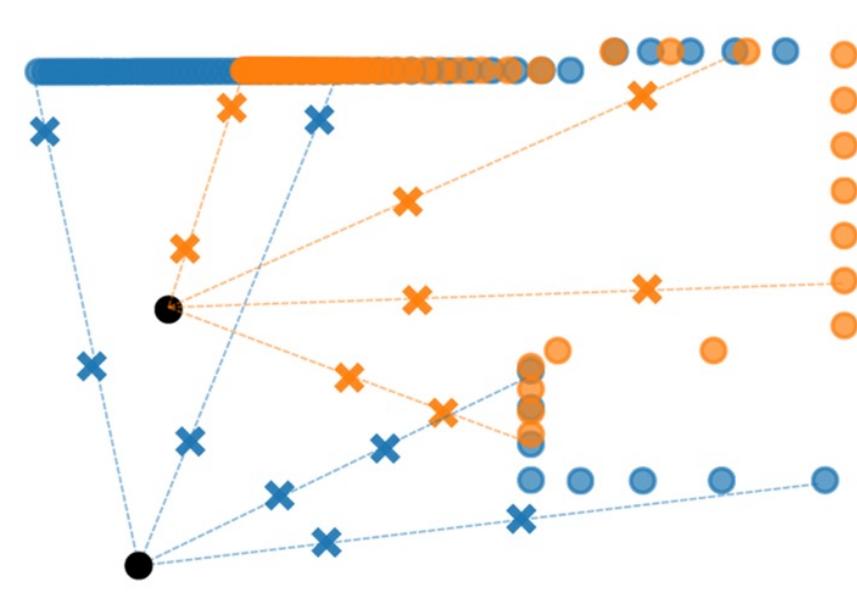
- Solve registration as an unsupervised “training” of neural network
  - Do not rely on ground truth data
  - Unsupervised loss measures registration quality
    - Minimize loss = encourage alignment



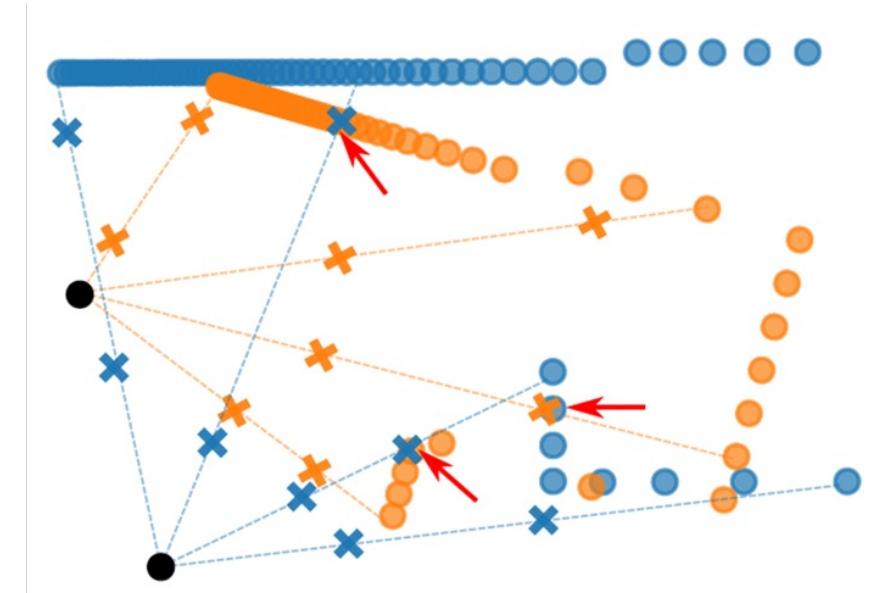


# DeepMapping Loss

- How do we design unsupervised loss to reflect the registration quality?
  - Utilize both points in observed point clouds and those in free-space
  - Free-space points in **one point cloud** not conflict with observed points in **another point cloud**



Correctly aligned  
No free-space inconsistency



Mis-aligned  
Free-space inconsistency



# DeepMapping Pipeline

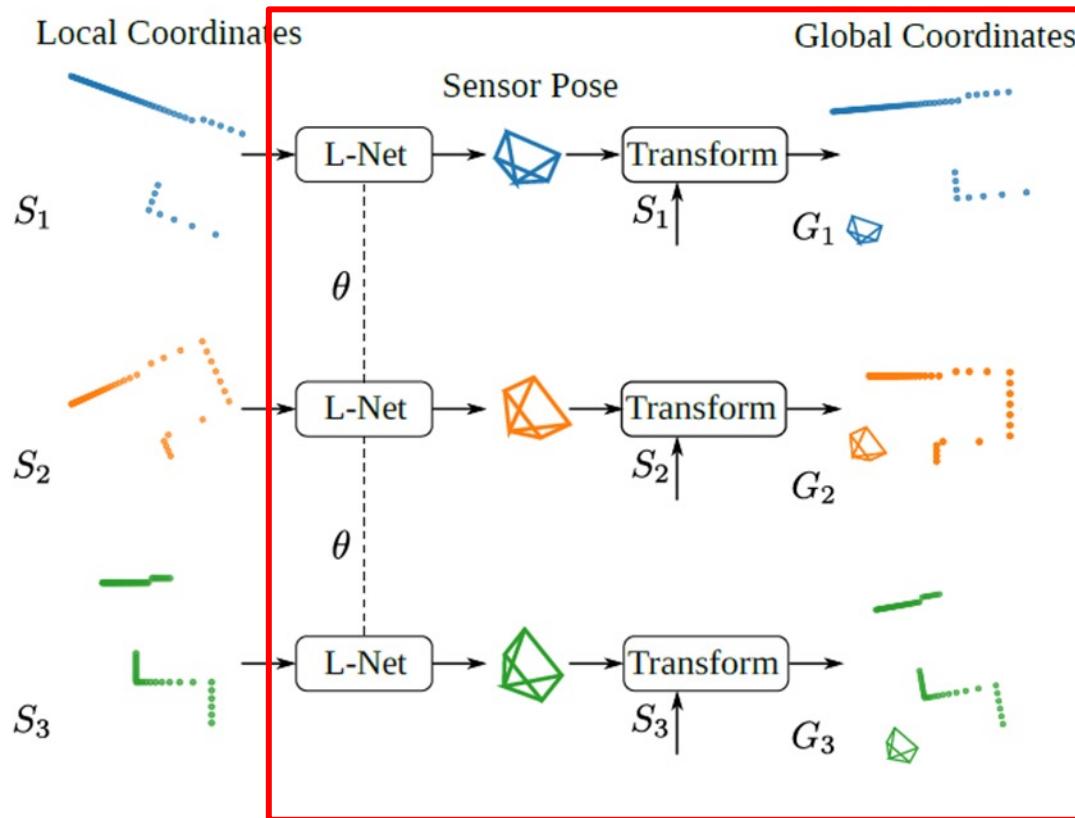
Local Coordinates



This leads us to the final DM pipeline. The input to DM is a set of point clouds in their local sensor coordinate systems.



# DeepMapping Pipeline: Localization Network



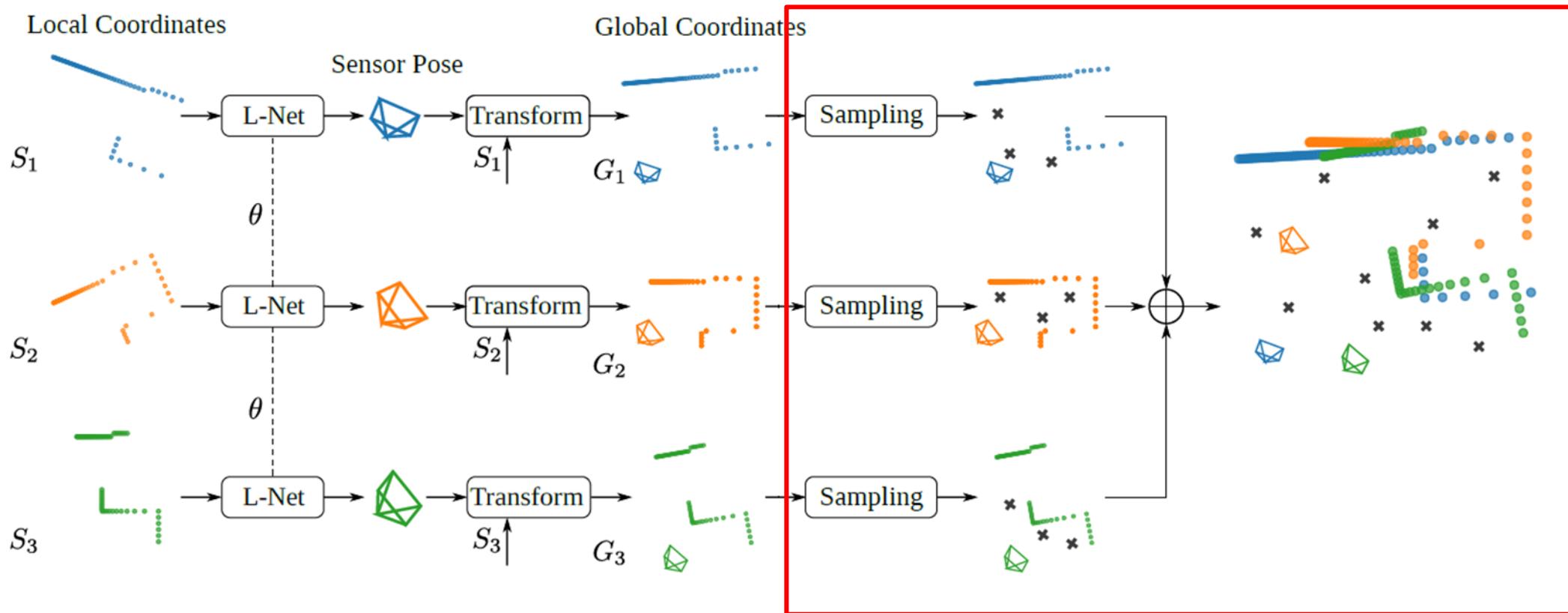
## Localization Network (L-Net)

$$f_{\theta} : S_i \mapsto T_i$$

We first use a localization network or L-Net that outputs the sensor pose for each point cloud and then transform all local point clouds into a common global frame.



# DeepMapping Pipeline: Free-Space Sampling



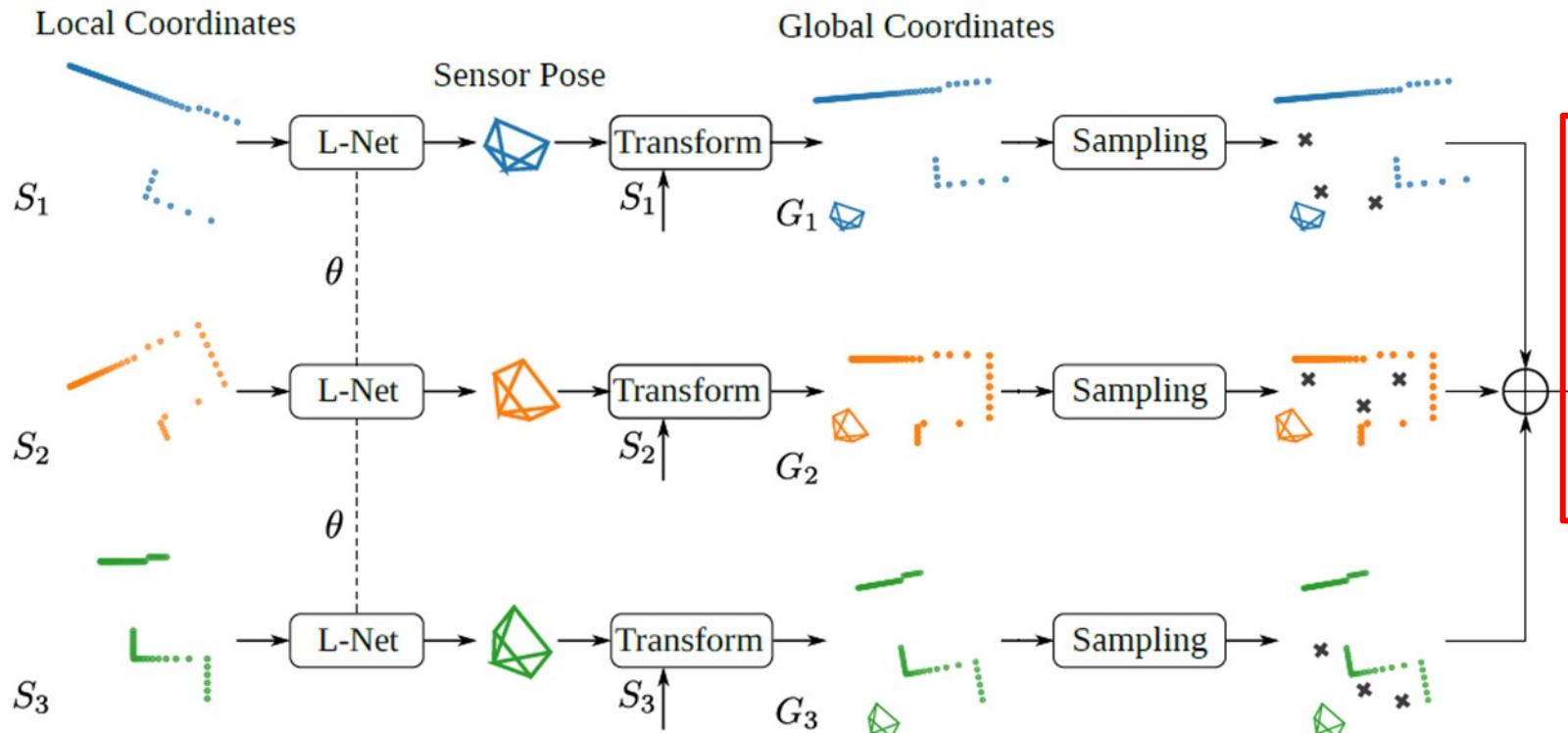
## Localization Network (L-Net)

$$f_{\theta} : S_i \mapsto T_i$$

In the global frame, we can sample free-space points from the line-of-sight at each scanning center.



# DeepMapping Pipeline: Occupancy Map Network



Occupancy Map Network (M-Net)

$$m_\phi : \mathbb{R}^D \rightarrow [0, 1]$$

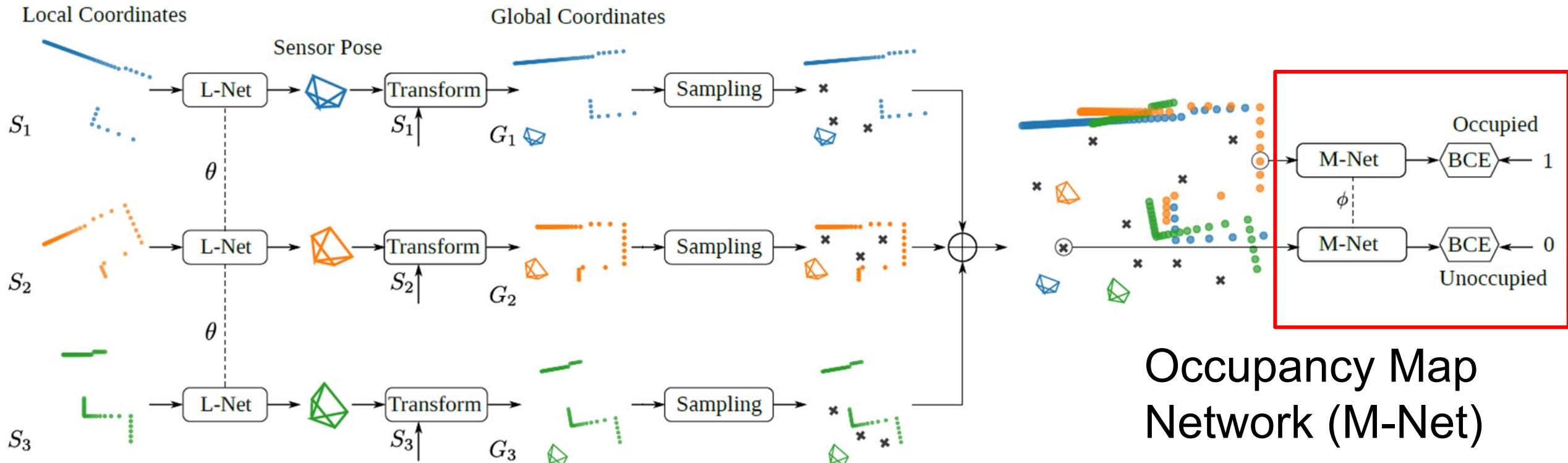
Localization Network (L-Net)

$$f_\theta : S_i \mapsto T_i$$

Then we introduce the second network Occupancy Map network M-Net, which is a binary classification network that predicts probabilities of input locations being occupied.



# DeepMapping Pipeline: Mapping Using Binary Classification Loss



Occupancy Map Network (M-Net)

$$m_\phi : \mathbb{R}^D \rightarrow [0, 1]$$

## Localization Network (L-Net)

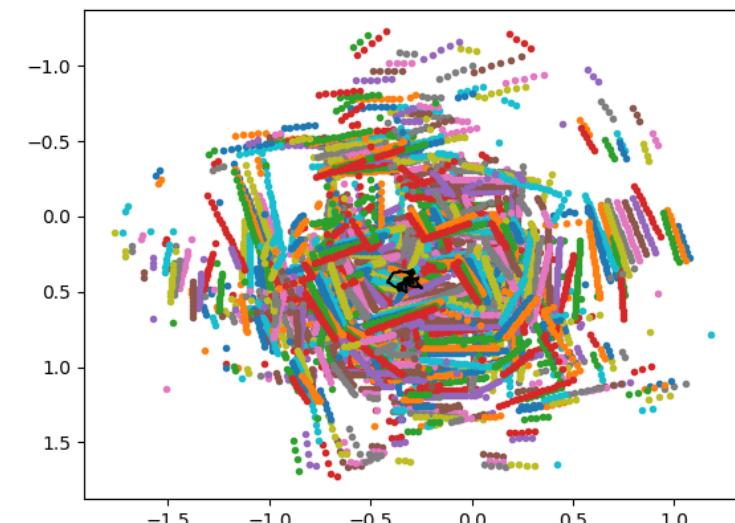
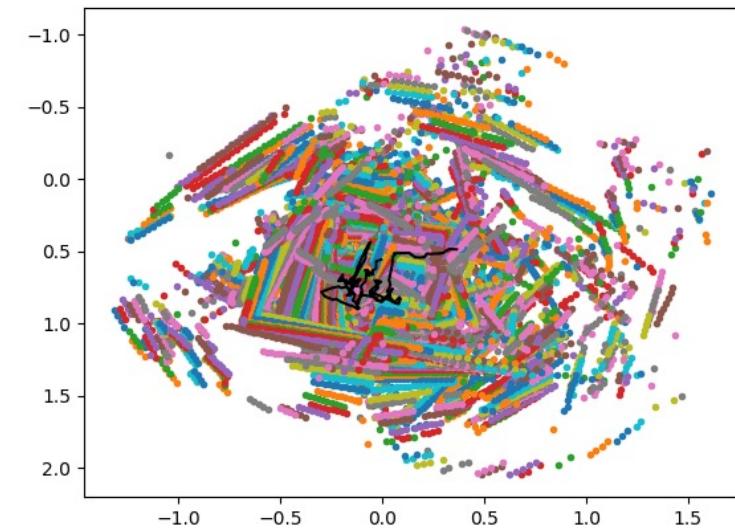
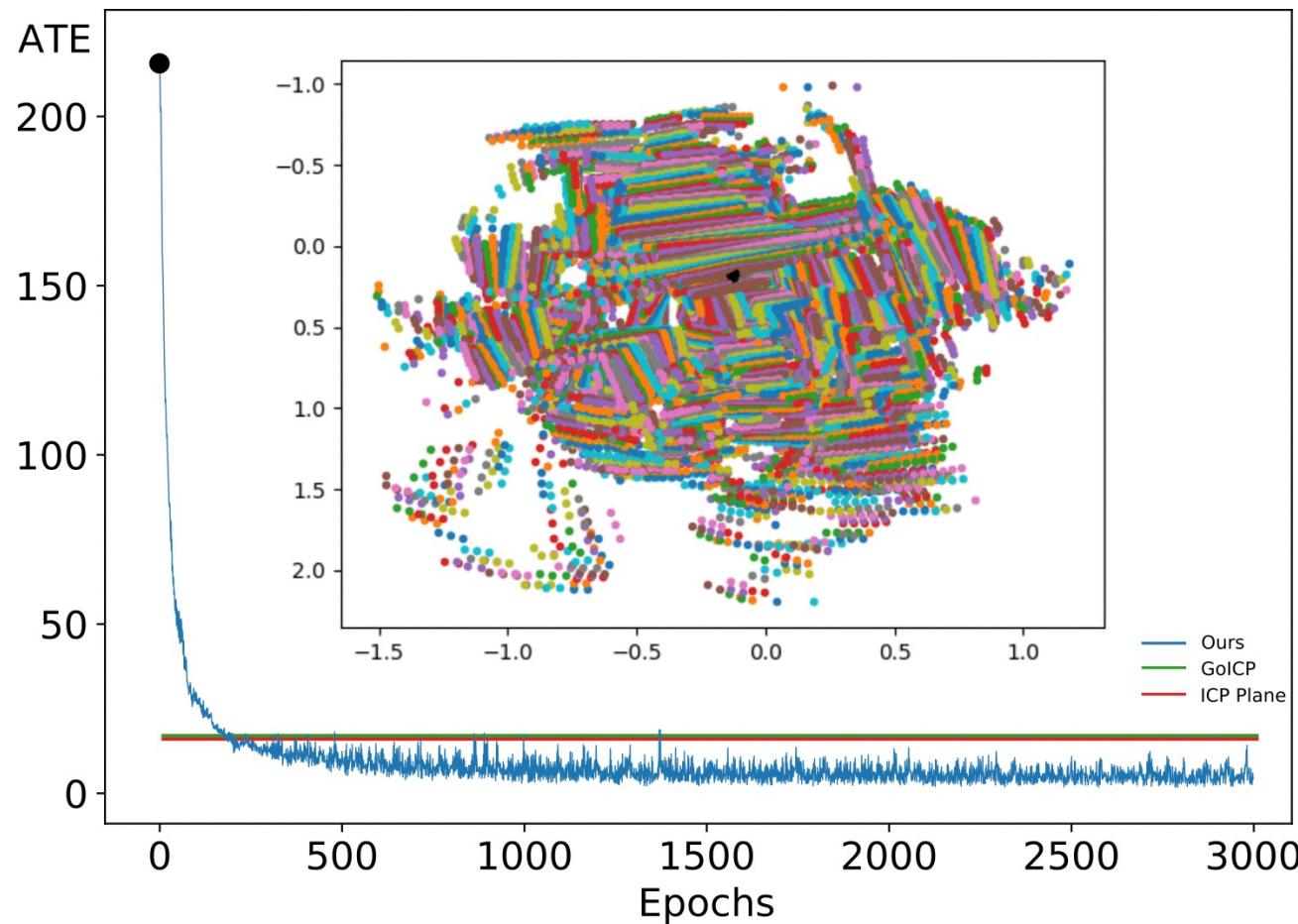
$$f_\theta : S_i \mapsto T_i$$

Those occupancy probabilities are used for computing the unsupervised loss, the BCE between predicted probability and occupancy labels. The DM is trained by minimizing the BCE loss and equivalently improve the registration quality.



# 2D Point Cloud Registration

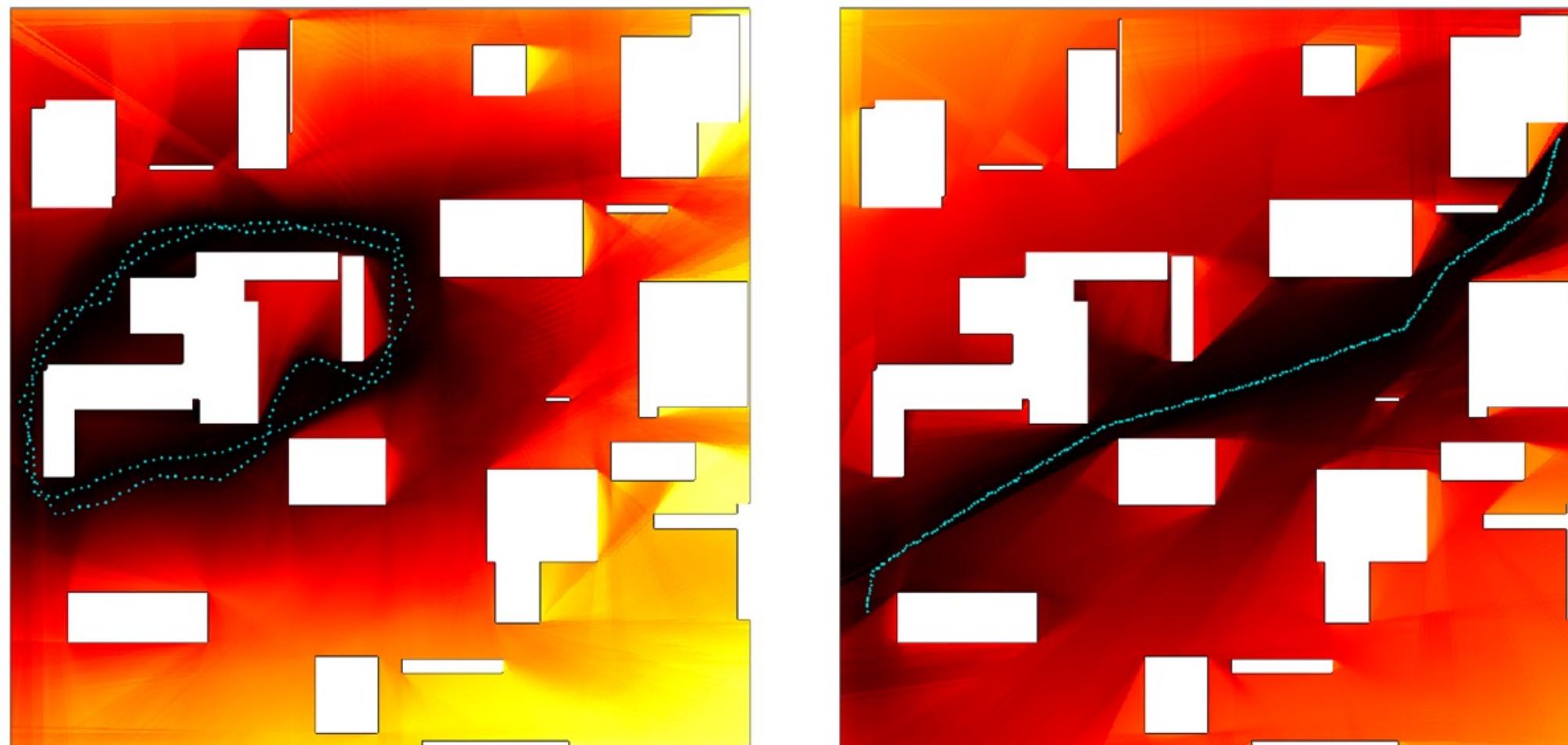
- Solve point cloud registration as “training” neural networks using unsupervised loss





# L-Net: Coarse Re-localization For Unseen Point Clouds

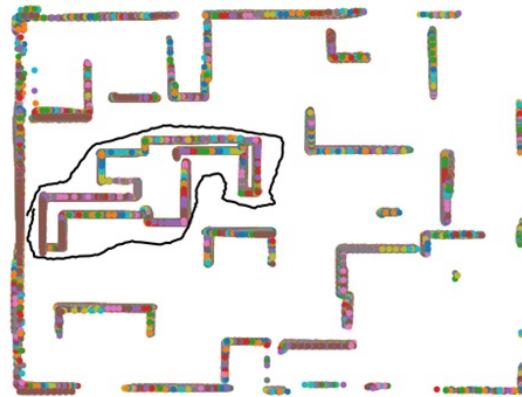
- Re-localization errors of two L-Nets “trained” on two trajectories
  - Darker is better



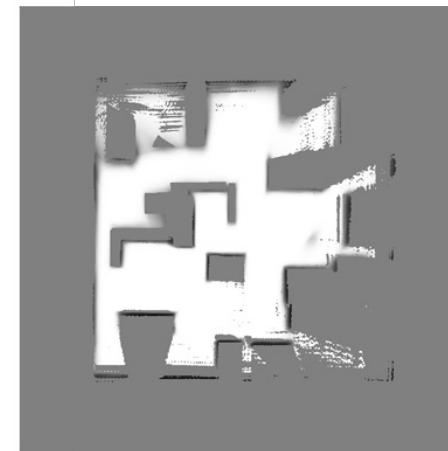


# M-Net: Continuous Occupancy Map Representation

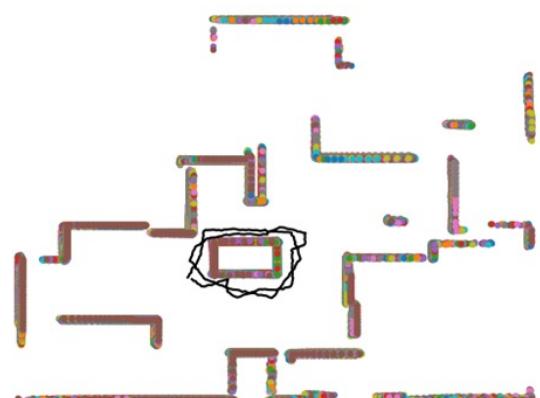
Aligned point clouds



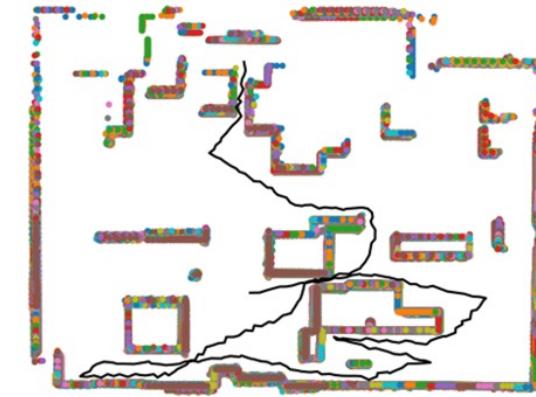
M-Net output



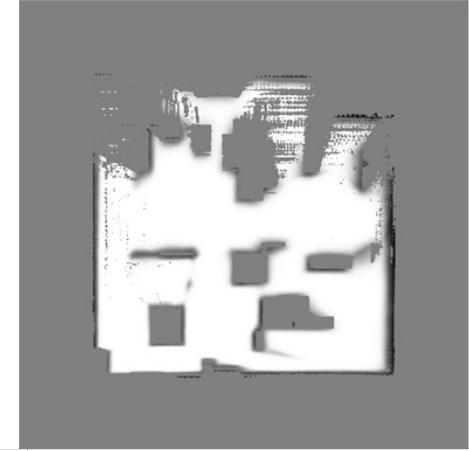
(Occupancy Map)



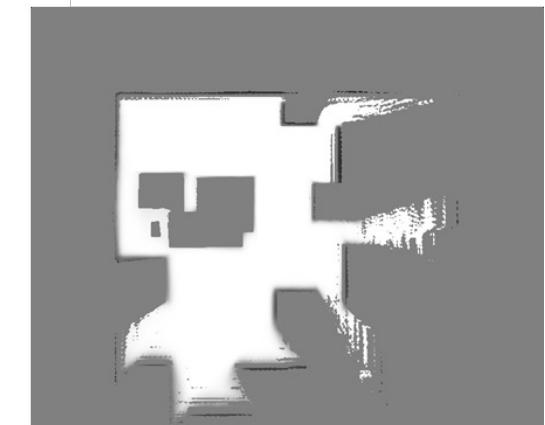
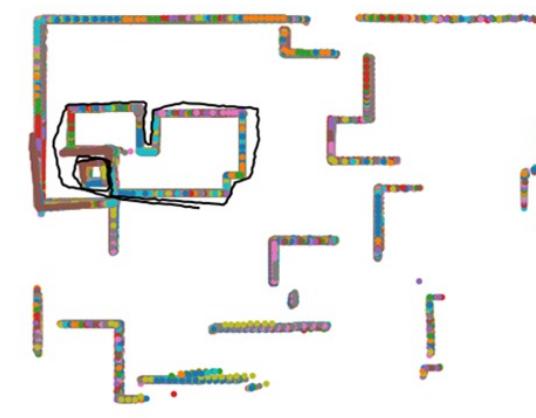
Aligned point clouds



M-Net output



(Occupancy Map)





## Next Week

- Adversarial robustness in perception
  - + White-box attack
  - + Black-box attack
  - + Universal adversarial perturbation (UAP)
  - + Physical adversarial examples
  - + Adversarial training and defense strategies
  - + Un-adversarial examples
  - + Beyond adversarial noise perturbation

+: know the concept



# References for Next Week

- Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." *arXiv preprint arXiv:1412.6572* (2014).
- Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.
- Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., & Swami, A. (2017, April). Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security* (pp. 506-519).
- Kurakin, A., Goodfellow, I. J., & Bengio, S. (2018). Adversarial examples in the physical world. In *Artificial intelligence safety and security* (pp. 99-112). Chapman and Hall/CRC.
- Moosavi-Dezfooli, S. M., Fawzi, A., Fawzi, O., & Frossard, P. (2017). Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1765-1773).
- Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.
- Salman, H., Ilyas, A., Engstrom, L., Vemprala, S., Madry, A., & Kapoor, A. (2021). Unadversarial examples: Designing objects for robust vision. *Advances in Neural Information Processing Systems*, 34, 15270-15284.
- Guo, Q., et al. (2020). Watch out! motion is blurring the vision of your deep neural networks. *Advances in Neural Information Processing Systems*, 33, 975-985.
- Li, Yiming, et al. "Fooling lidar perception via adversarial trajectory perturbation." *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021.