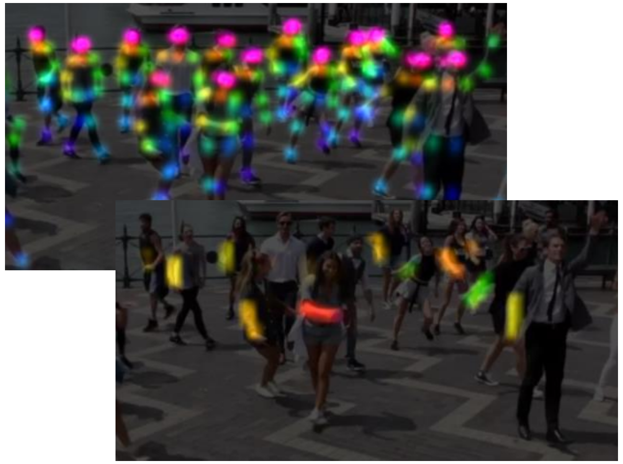
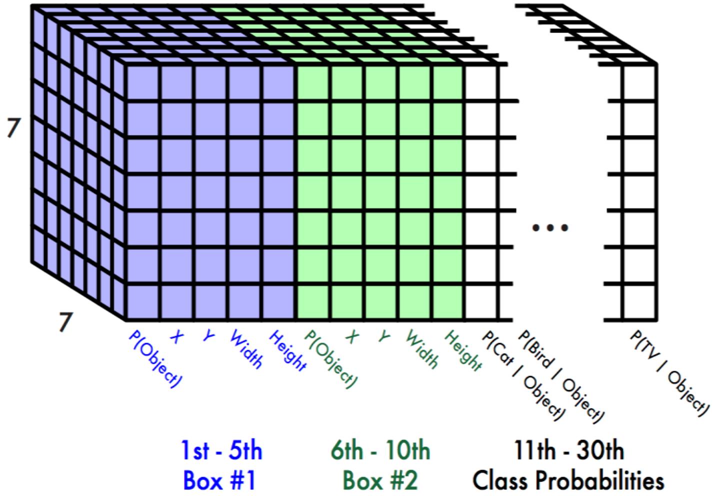
3D Point Processing

James Hays

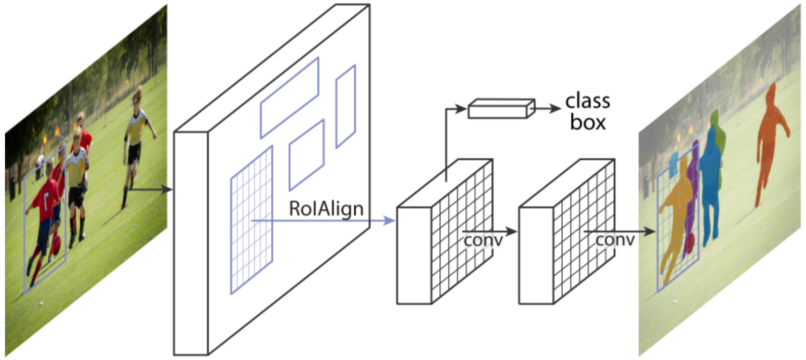
Recap – Structured Output from Deep Networks



Convolutional Pose Machines and follow up works



YOLO, SSD, and "one stage" object detectors



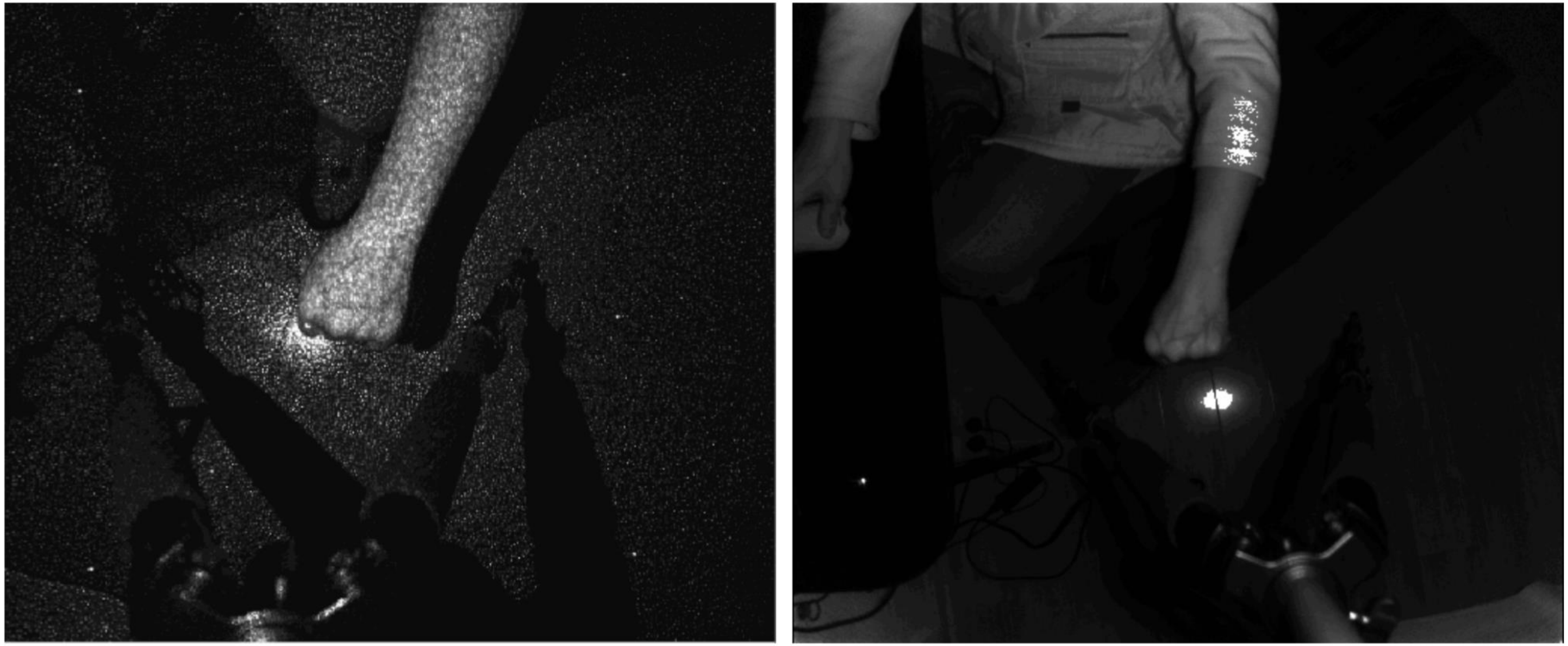
Mask R-CNN and "two stage" object detectors

A lot of machine learning tools, such as convolutional networks, don't naturally handle tasks with arbitrary numbers of outputs. These are a few clever methods, typical of the literature as a whole, to work around this.

Outline

- How do we measure 3D points?
- How do we make decisions about point clouds?
    - PointNet – orderless point processing
    - VoxelNet – voxel-based point processing
    - PointPillars – bird’s eye view point processing
        - Exploiting Visibility for 3D Object Detection
    - LaserNet – range image point processing
- PseudoLidar – Bird’s eye view depth map processing

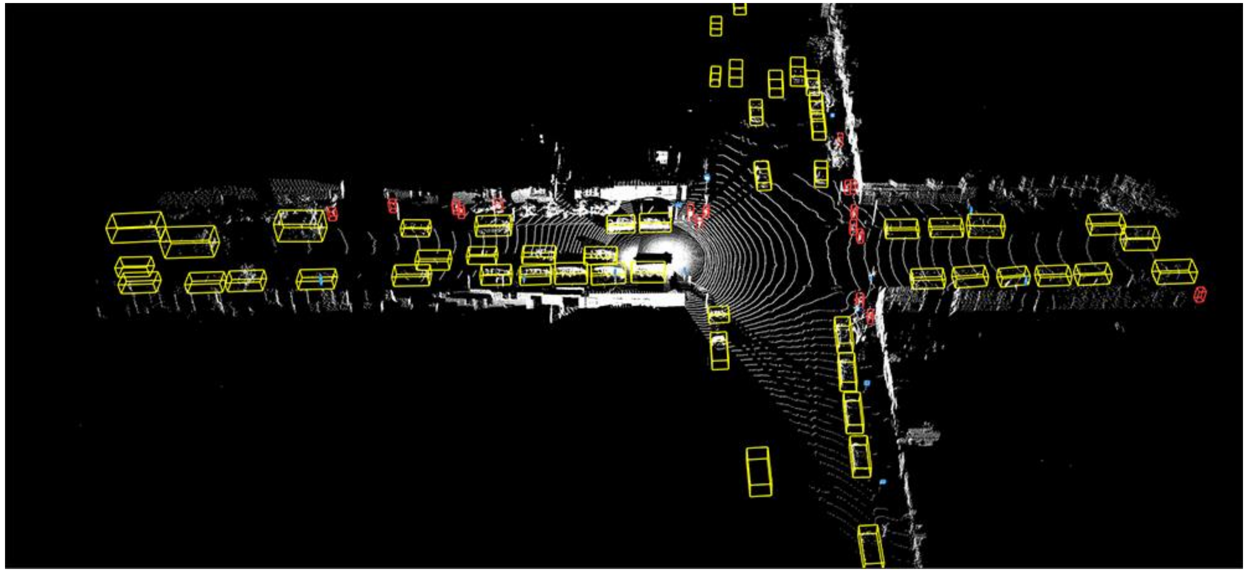
Kinect V1 and V2



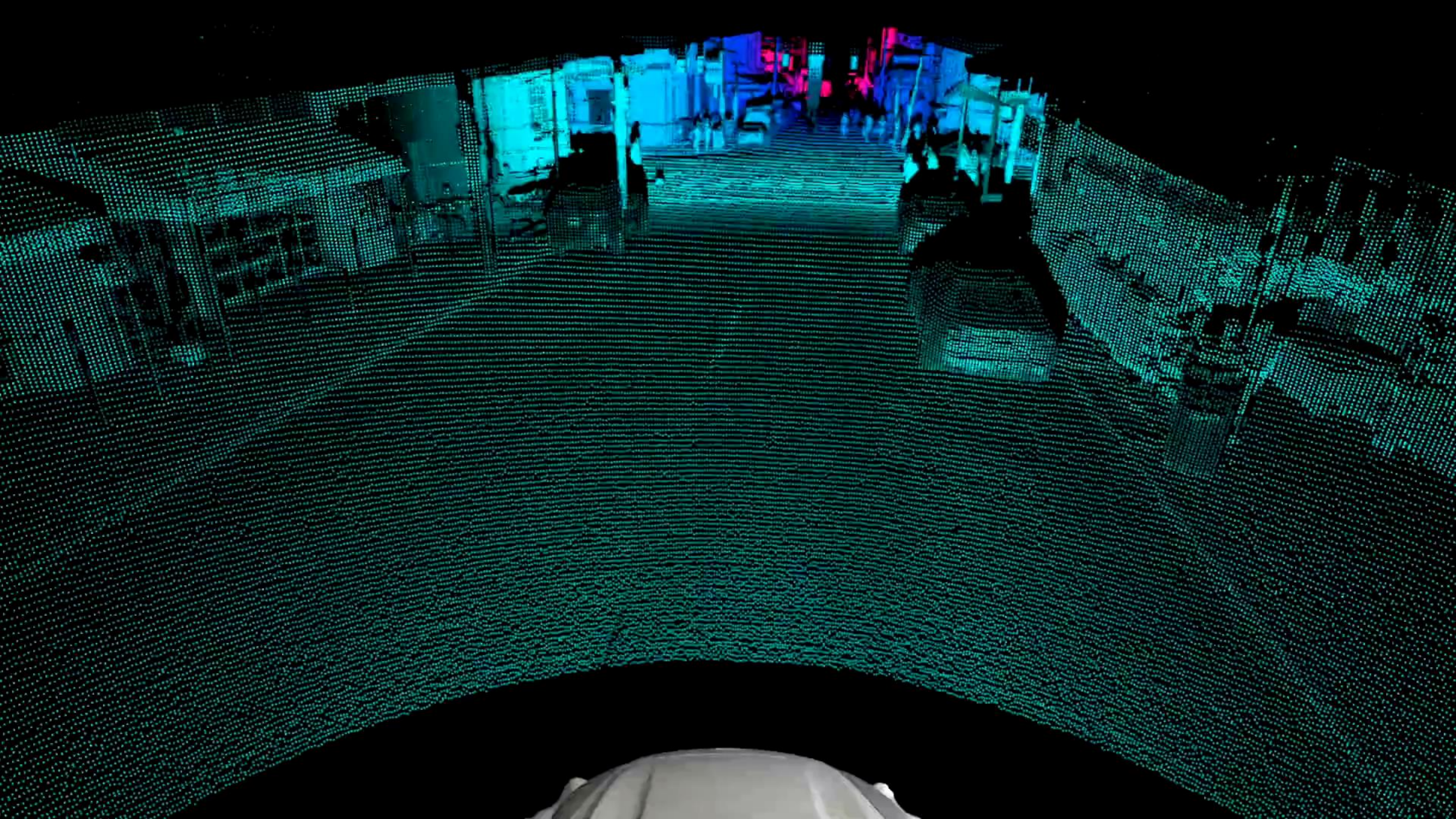
Infrared images of Kinect V1 structured light pattern and Kinect V2 time of flight pattern. Credit "Lightweight Algorithms for Depth Sensor Equipped Embedded Devices" by Henry Zhong

Lidar overview

Lidar overview



Source: Waymo Open Dataset



Outline

- What is lidar?
- How do we make decisions about point clouds?
  - PointNet – orderless point processing
  - VoxelNet – voxel-based point processing
  - PointPillars – bird’s eye view point processing
    - Exploiting Visibility for 3D Object Detection
  - LaserNet – range image point processing
- PseudoLidar – Bird’s eye view depth map processing

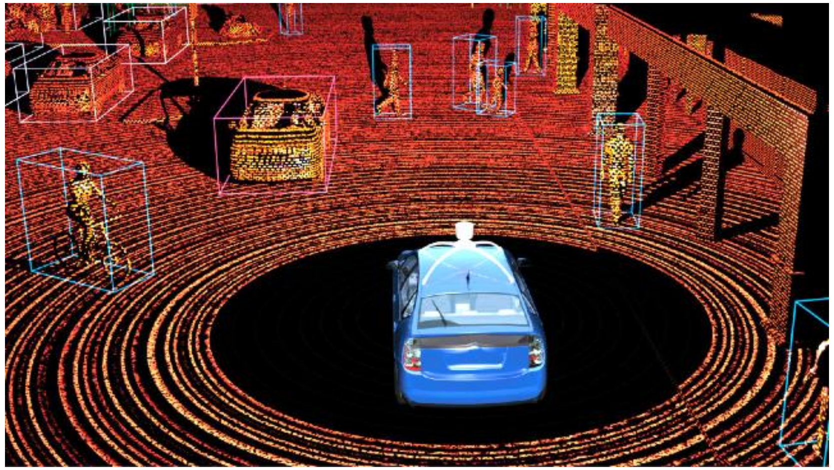
PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation

Charles R. Qi* Hao Su* Kaichun Mo Leonidas J. Guibas

Stanford University

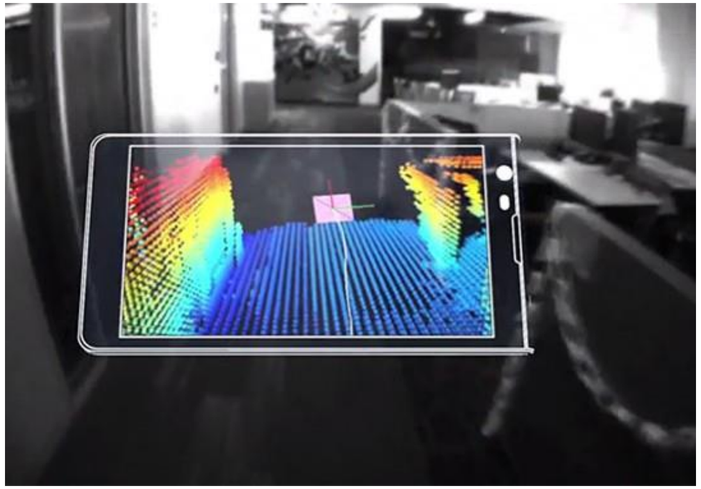
Big Data + Deep Representation Learning

Robot Perception



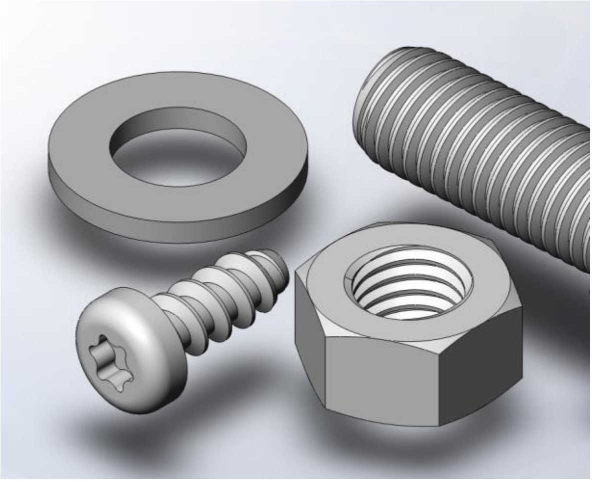
source: Scott J Grunewald

Augmented Reality



source: Google Tango

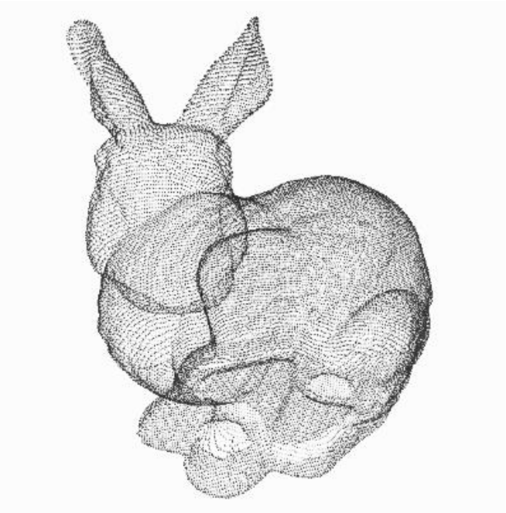
Shape Design



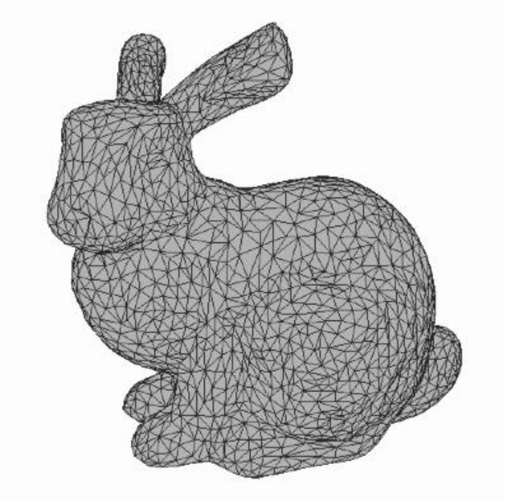
source: solidsolutions

Need for 3D Deep Learning!

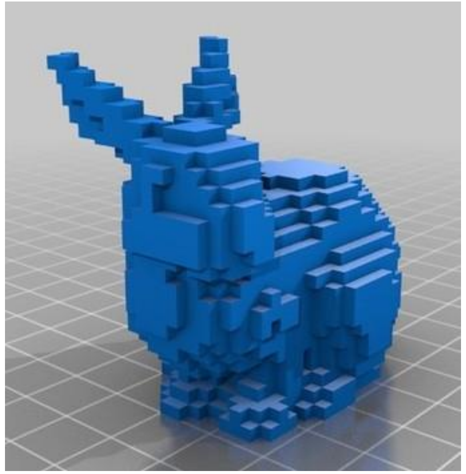
3D Representations



Point Cloud



Mesh



Volumetric

Projected View
RGB(D)

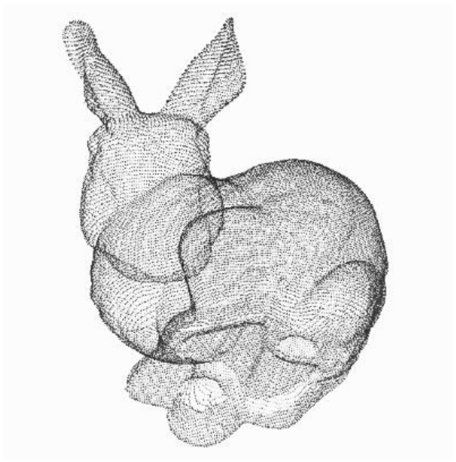
...

3D Representation: Point Cloud

Point cloud is close to raw sensor data

LiDAR

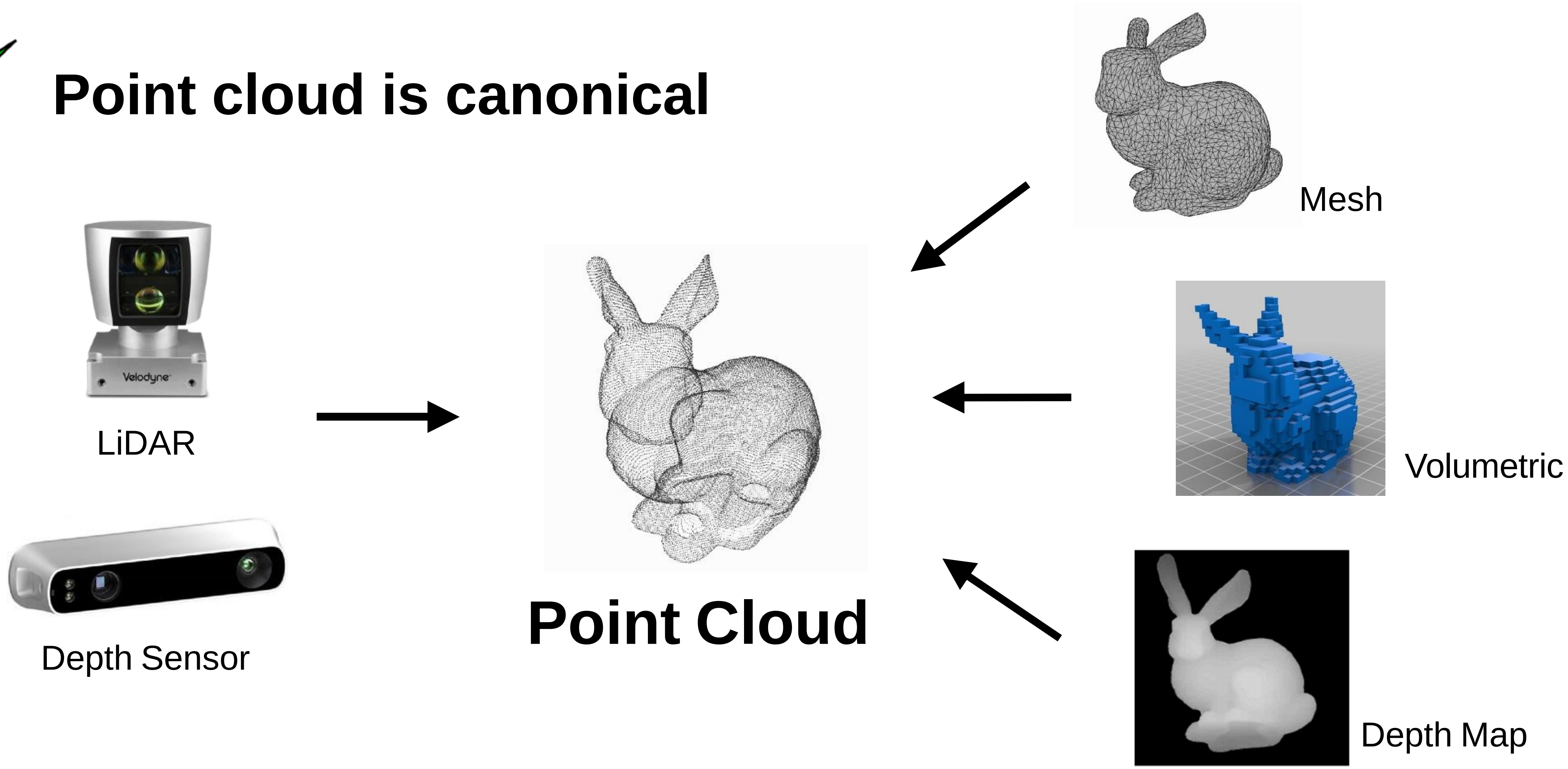
Depth Sensor



Point Cloud

3D Representation: Point Cloud

- Point cloud is close to raw sensor data
- Point cloud is canonical



Previous Works

Most existing point cloud features are **handcrafted** towards specific tasks

Feature Name	Supports Texture / Color	Local / Global / Regional	Best Use Case
PFH	No	L	
FPFH	No	L	2.5D Scans (Pseudo single position range images)
VFH	No	G	Object detection with basic pose estimation
CVFH	No	R	Object detection with basic pose estimation, detection of partial objects
RIFT	Yes	L	Real world 3D-Scans with no mirror effects. RIFT is vulnerable against flipping.

Source: https://github.com/PointCloudLibrary/pcl/wiki/Overview-and-Comparison-of-Features

Previous Works

Point cloud is **converted to other representations** before it's fed to a deep neural network

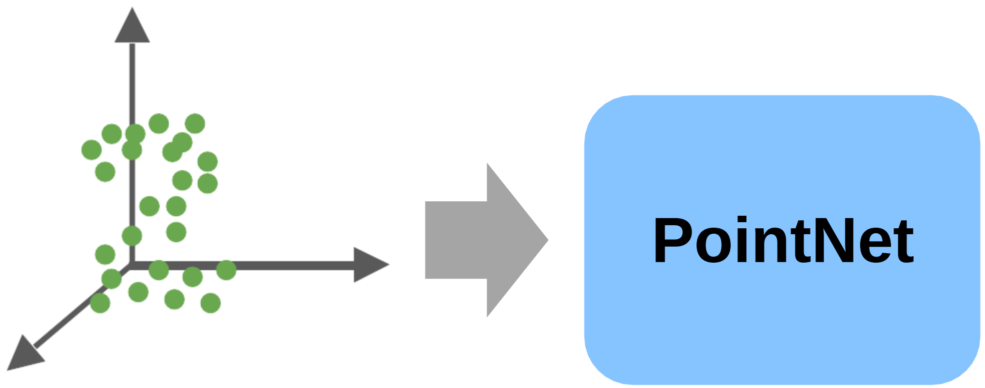
Conversion	Deep Net
Voxelization	3D CNN
Projection/Rendering	2D CNN
Feature extraction	Fully Connected

Research Question:

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

Our Work: PointNet

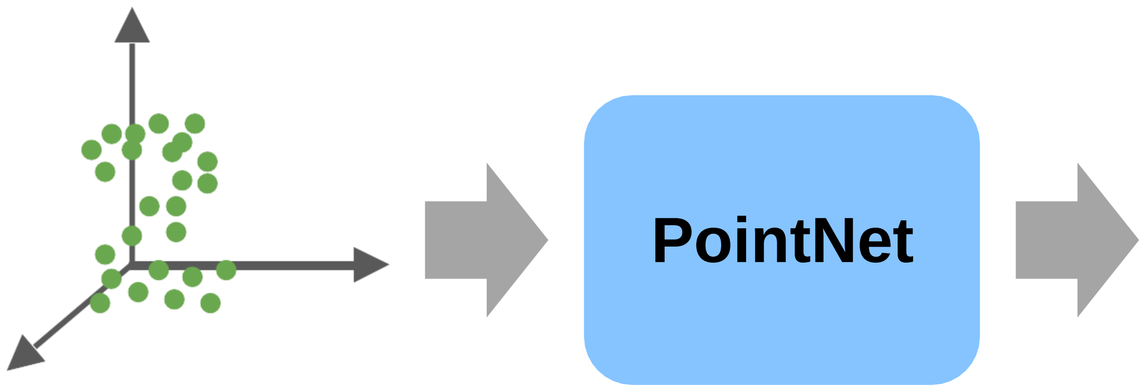
End-to-end learning for **scattered, unordered** point data



Our Work: PointNet

End-to-end learning for **scattered, unordered** point data

Unified framework for various tasks



Object Classification

Object Part Segmentation

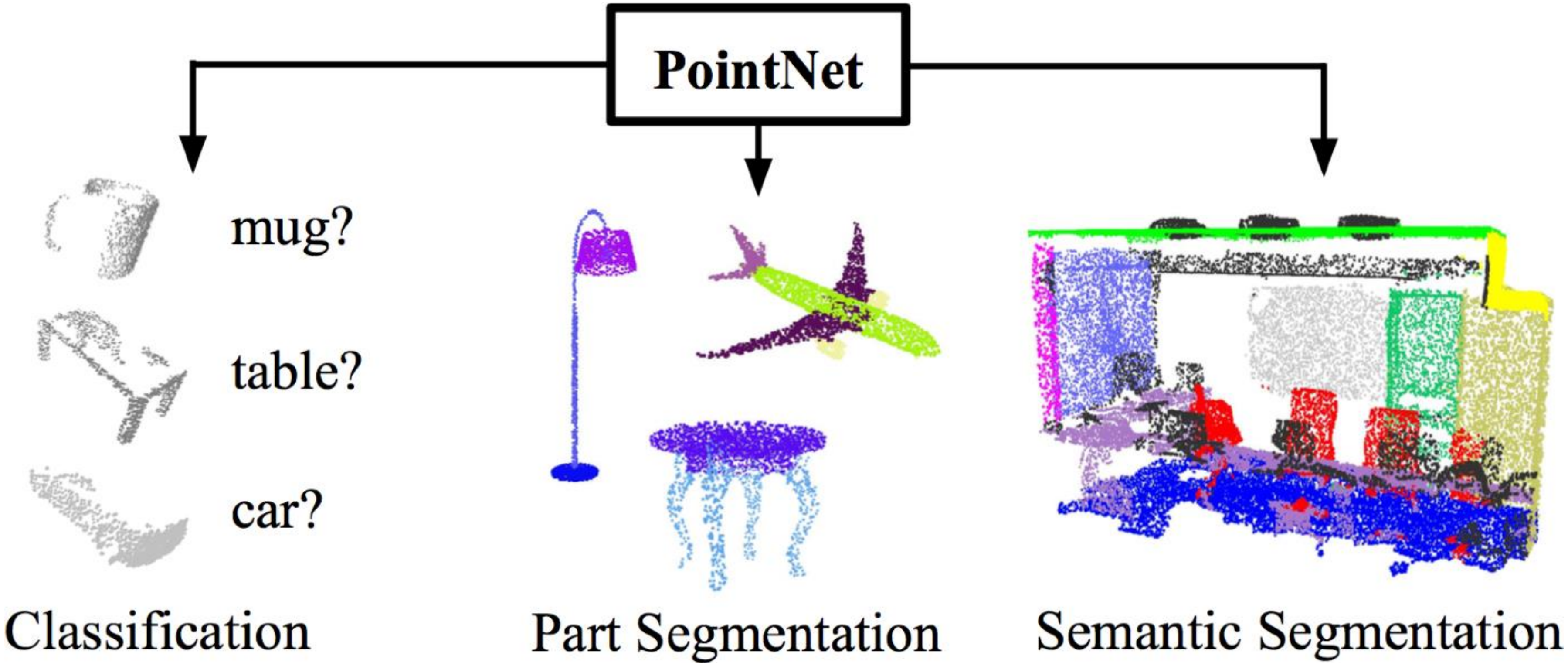
Semantic Scene Parsing

...

Our Work: PointNet

End-to-end learning for **scattered, unordered** point data

Unified framework for various tasks



Challenges

Unordered point set as input

Model needs to be invariant to N! permutations.

Invariance under geometric transformations

Point cloud rotations should not alter classification results.

Challenges

Unordered point set as input

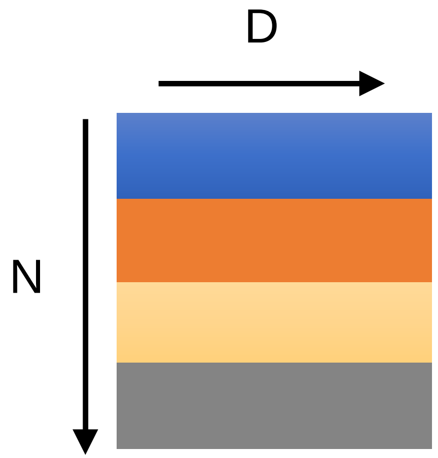
Model needs to be invariant to N! permutations.

Invariance under geometric transformations

Point cloud rotations should not alter classification results.

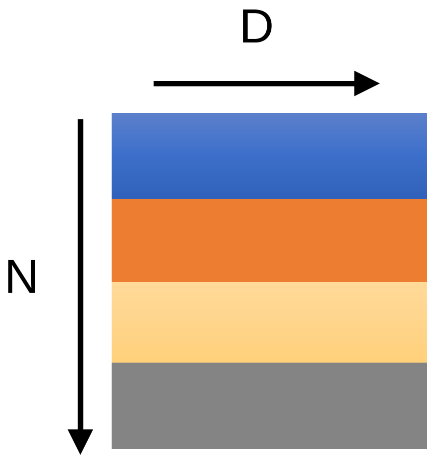
Unordered Input

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

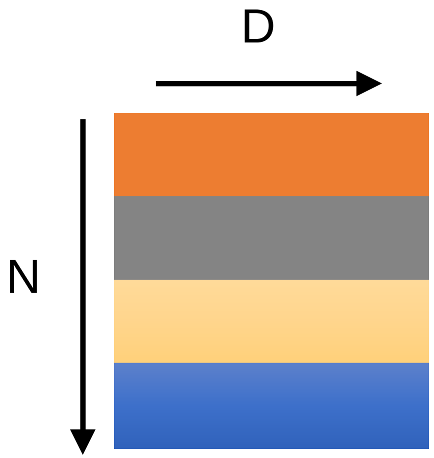


Unordered Input

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

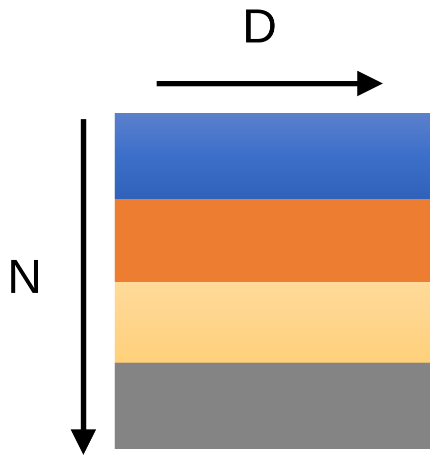


represents the same **set** as

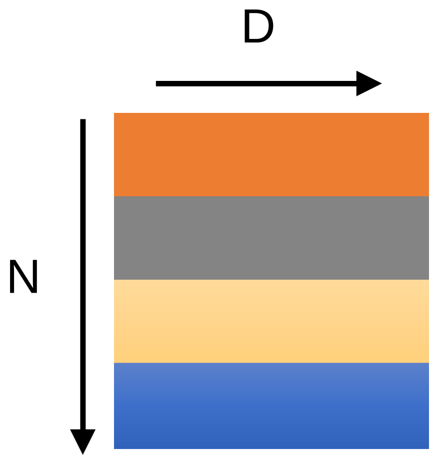


Unordered Input

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



represents the same **set** as



Model needs to be invariant to N! permutations

Permutation Invariance: Symmetric Function

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

Permutation Invariance: Symmetric Function

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

Examples:

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

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

$\ldots$

Permutation Invariance: Symmetric Function

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

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 family of symmetric functions by neural networks?

Permutation Invariance: Symmetric Function

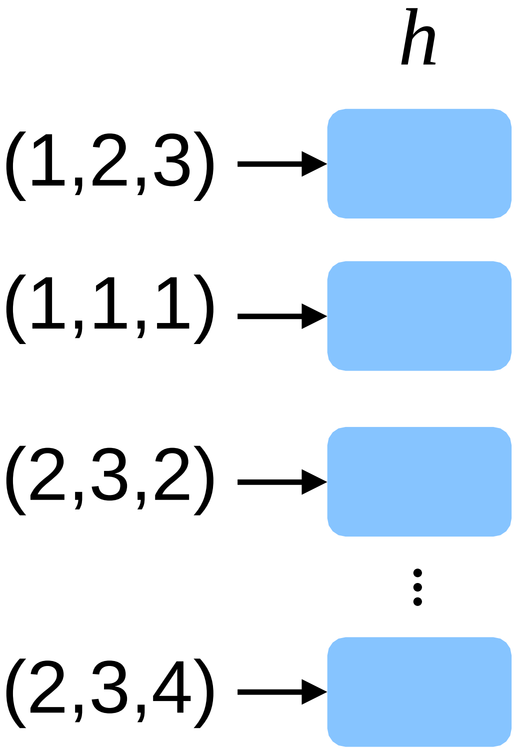
Observe:

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

Permutation Invariance: Symmetric Function

Observe:

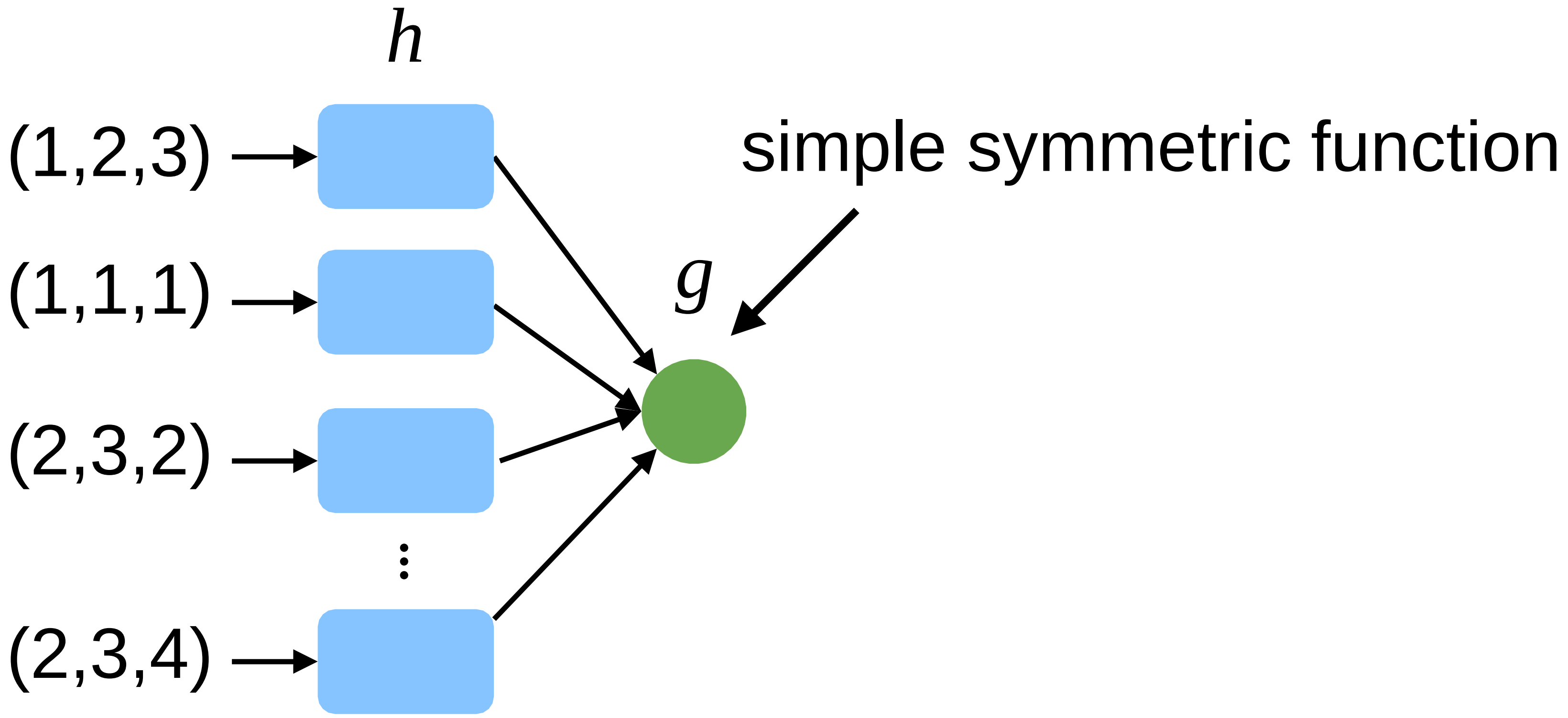
$f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n))$ is symmetric if g is symmetric



Permutation Invariance: Symmetric Function

Observe:

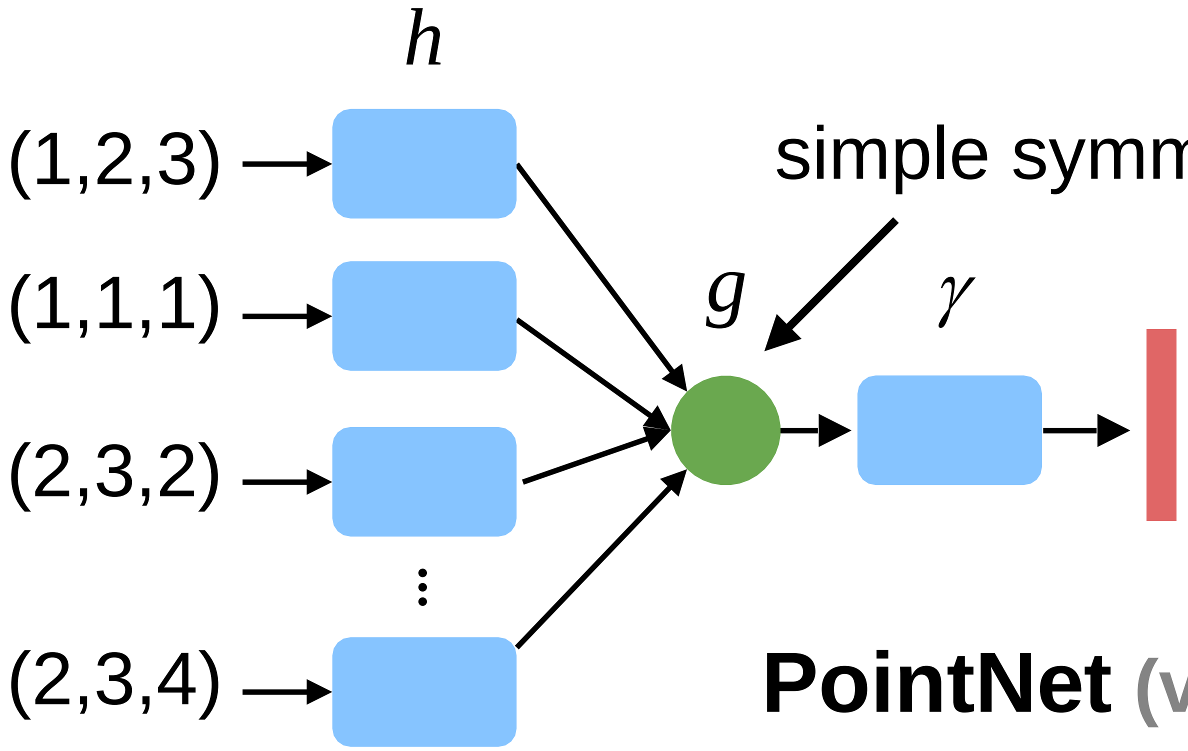
$f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n))$ is symmetric if g is symmetric



Permutation Invariance: Symmetric Function

Observe:

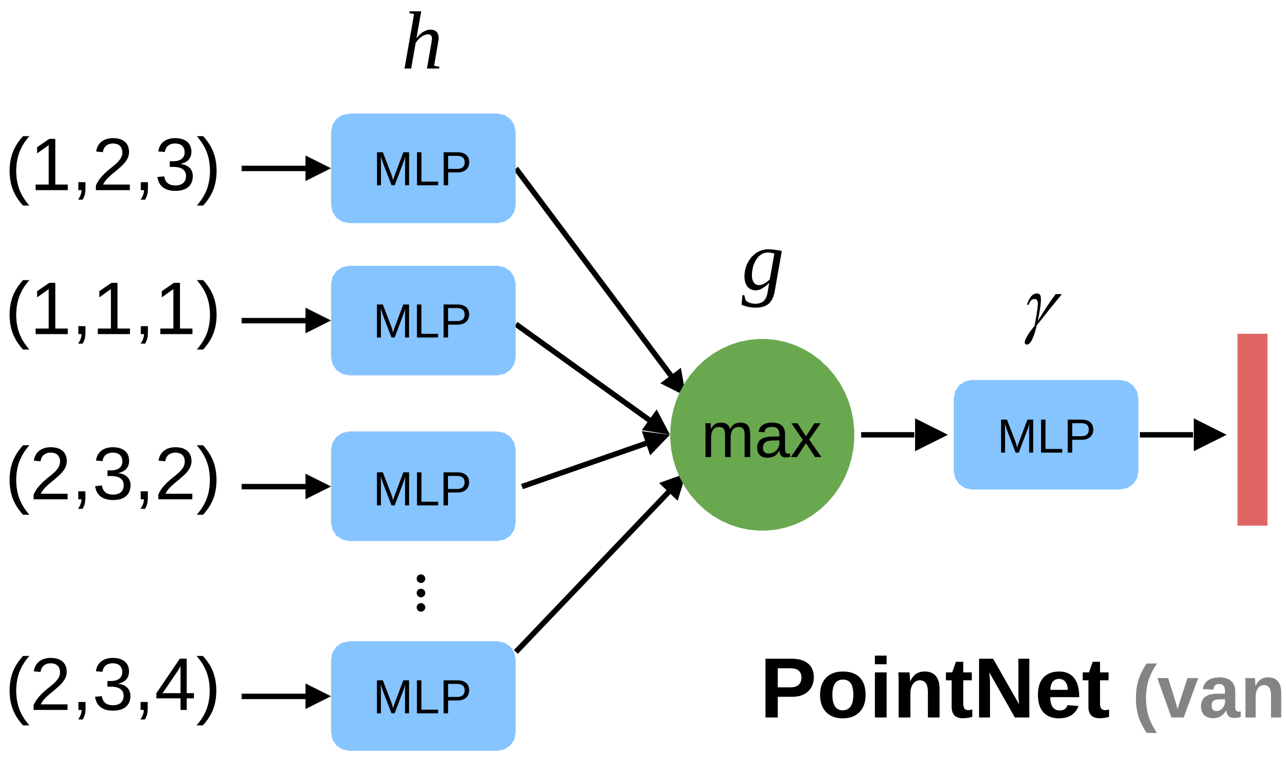
$$f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n))$$ is symmetric if g is symmetric



PointNet **(vanilla)**

Basic PointNet Architecture

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



PointNet (vanilla)

Challenges

Unordered point set as input

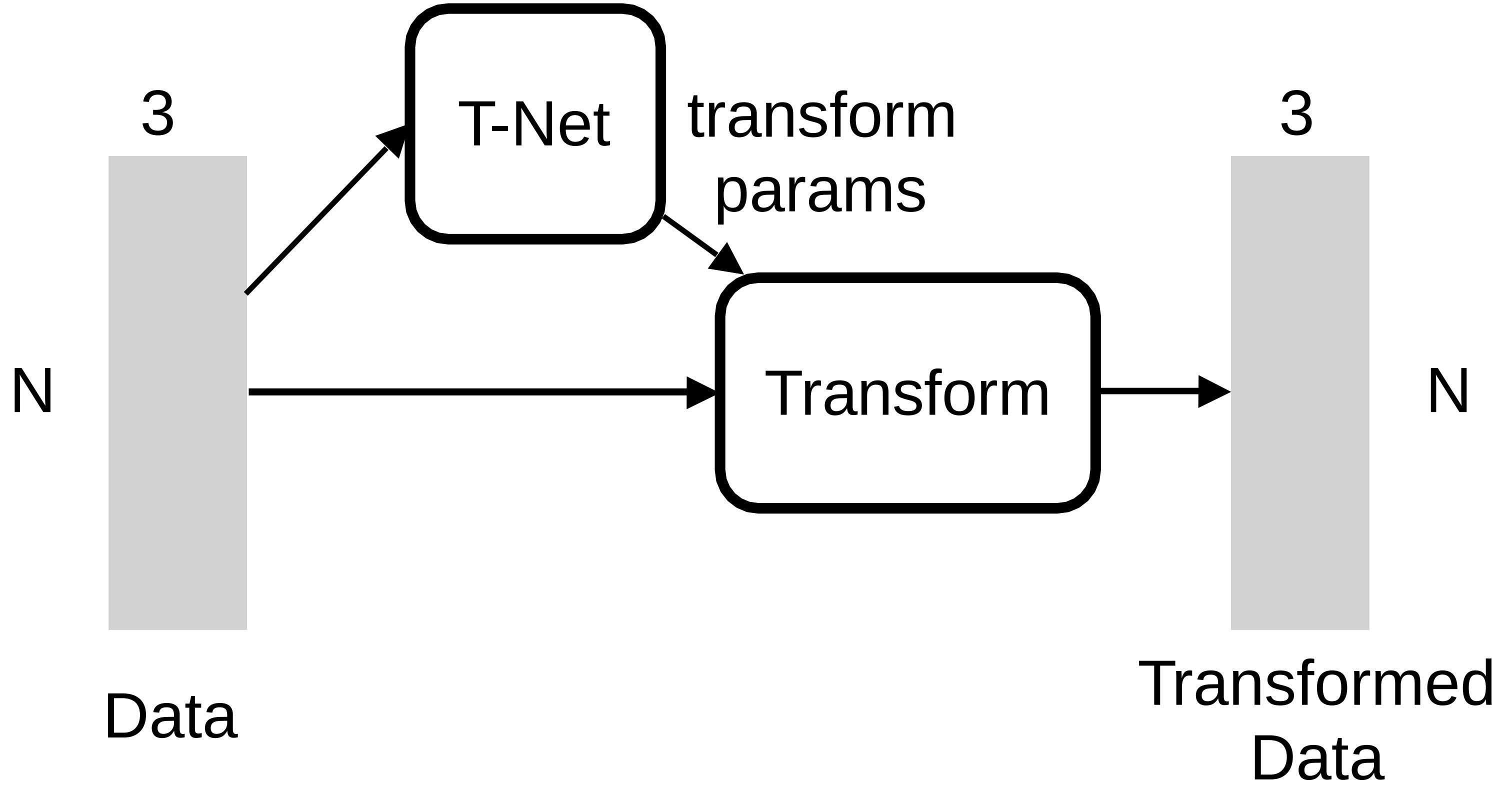
Model needs to be invariant to N! permutations.

Invariance under geometric transformations

Point cloud rotations should not alter classification results.

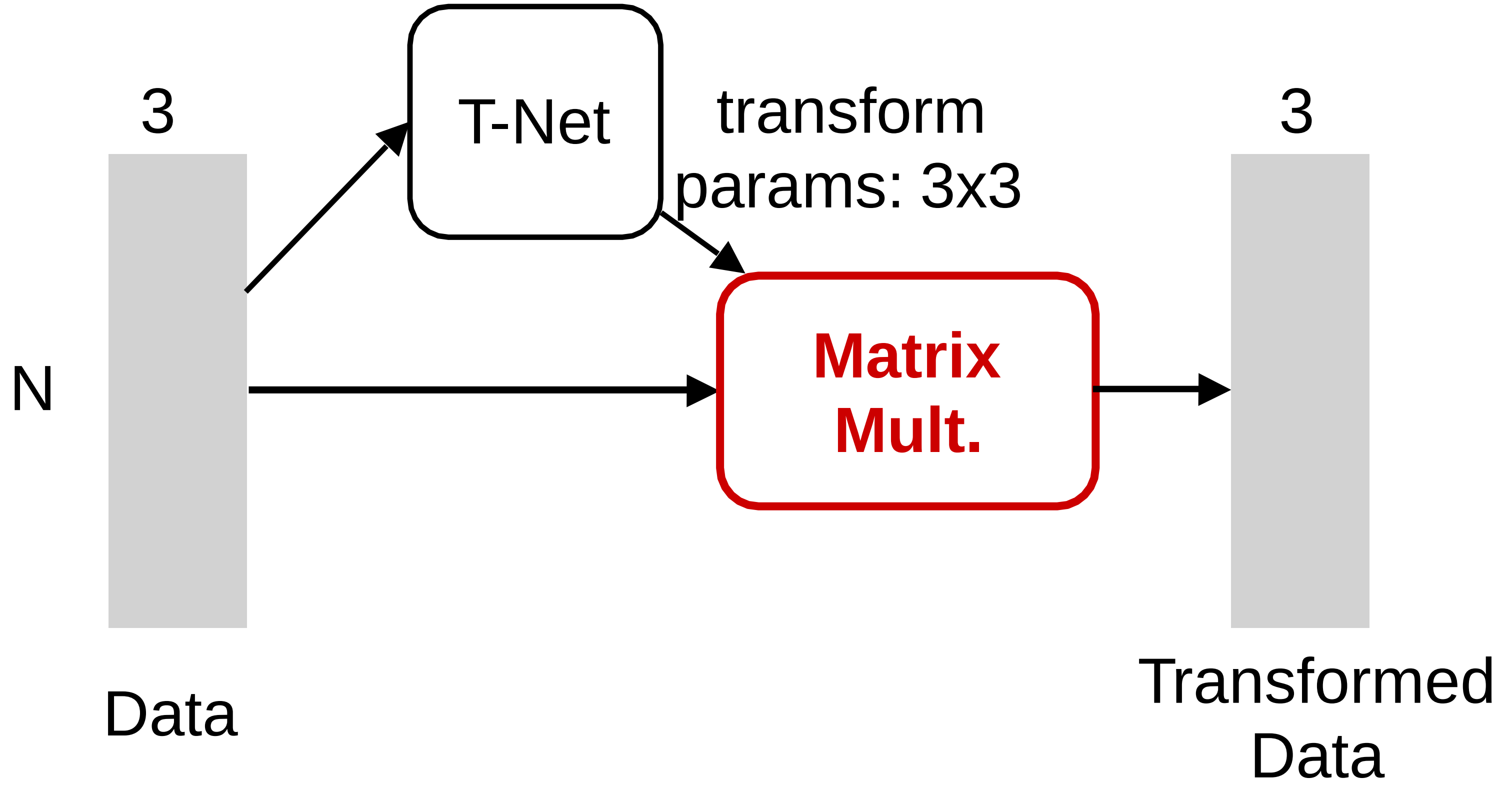
Input Alignment by Transformer Network

Idea: Data dependent transformation for automatic alignment

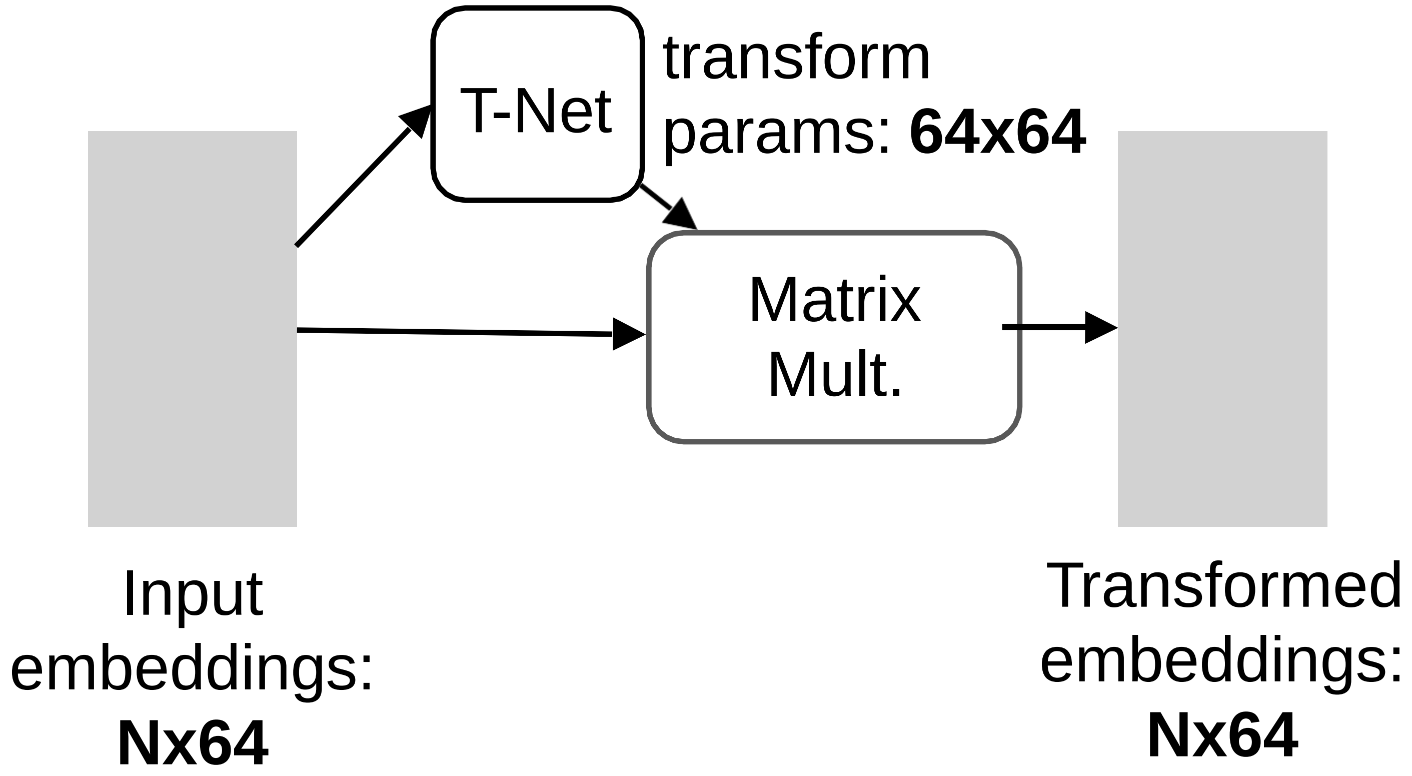


Input Alignment by Transformer Network

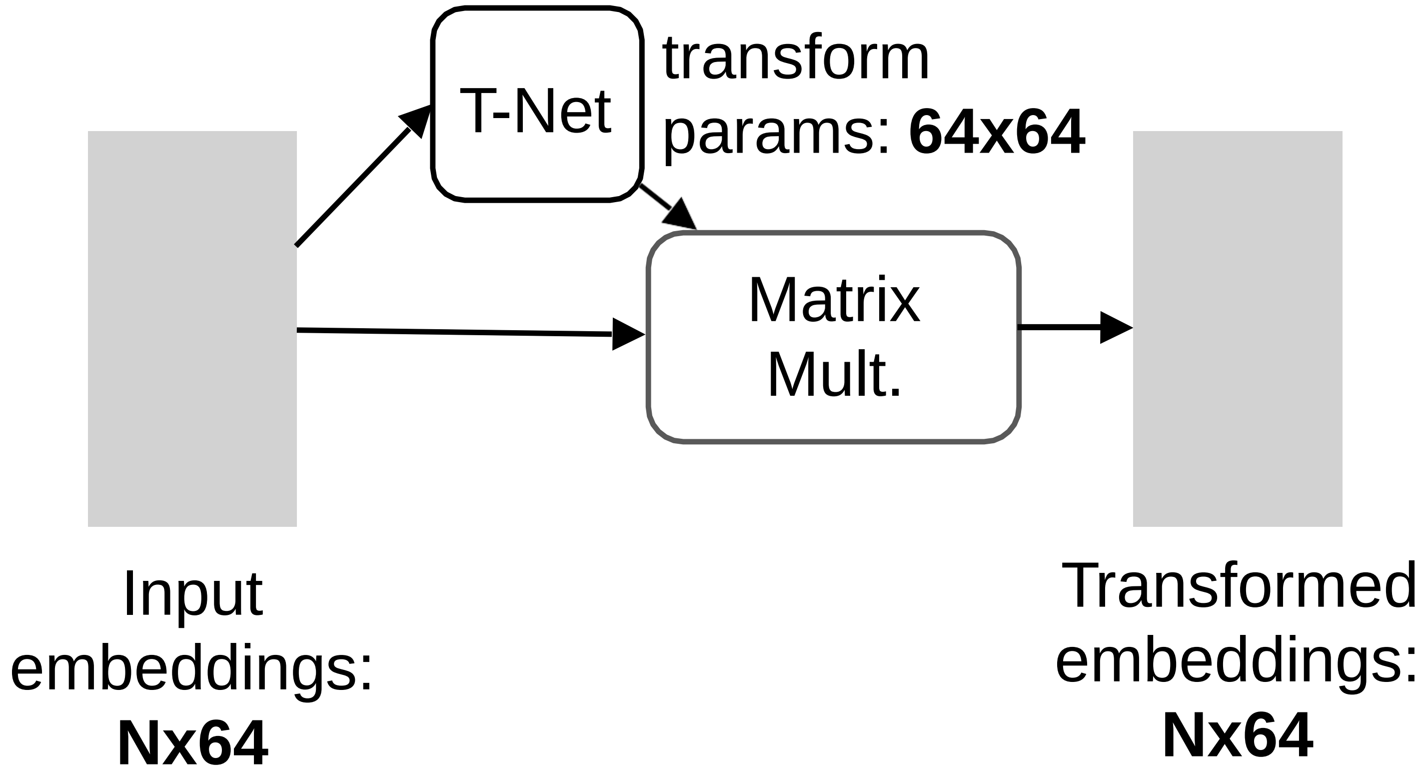
The transformation is just matrix multiplication!



Embedding Space Alignment



Embedding Space Alignment



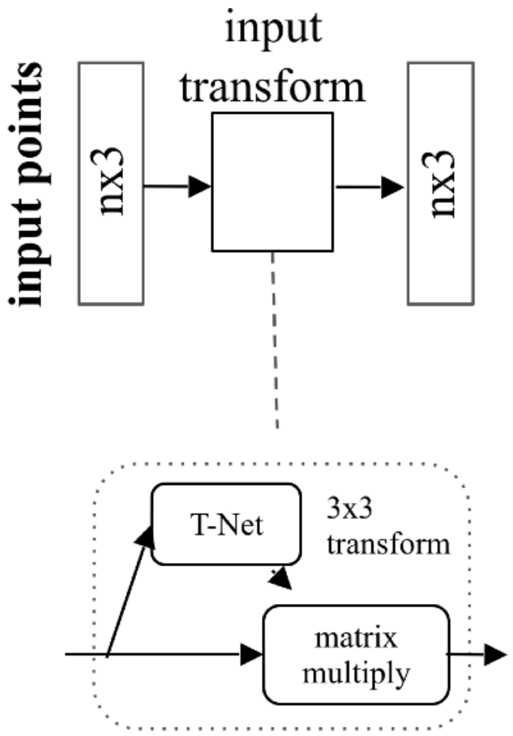
Regularization:

Transform matrix A 64x64 close to orthogonal:

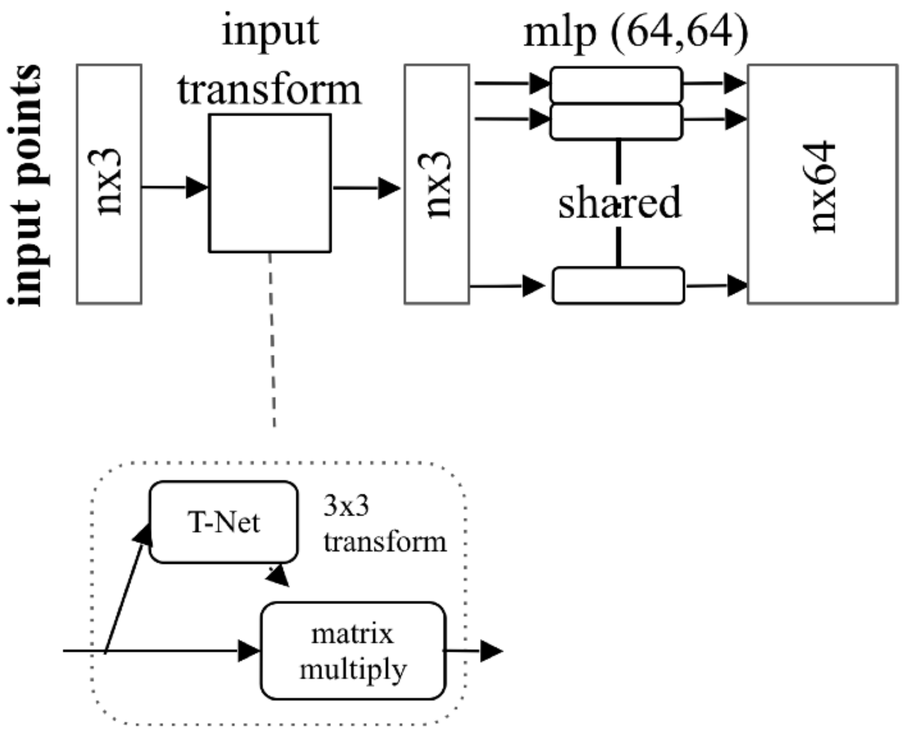
$$L_{reg} = \|I - AA^T\|_F^2$$

PointNet Classification Network

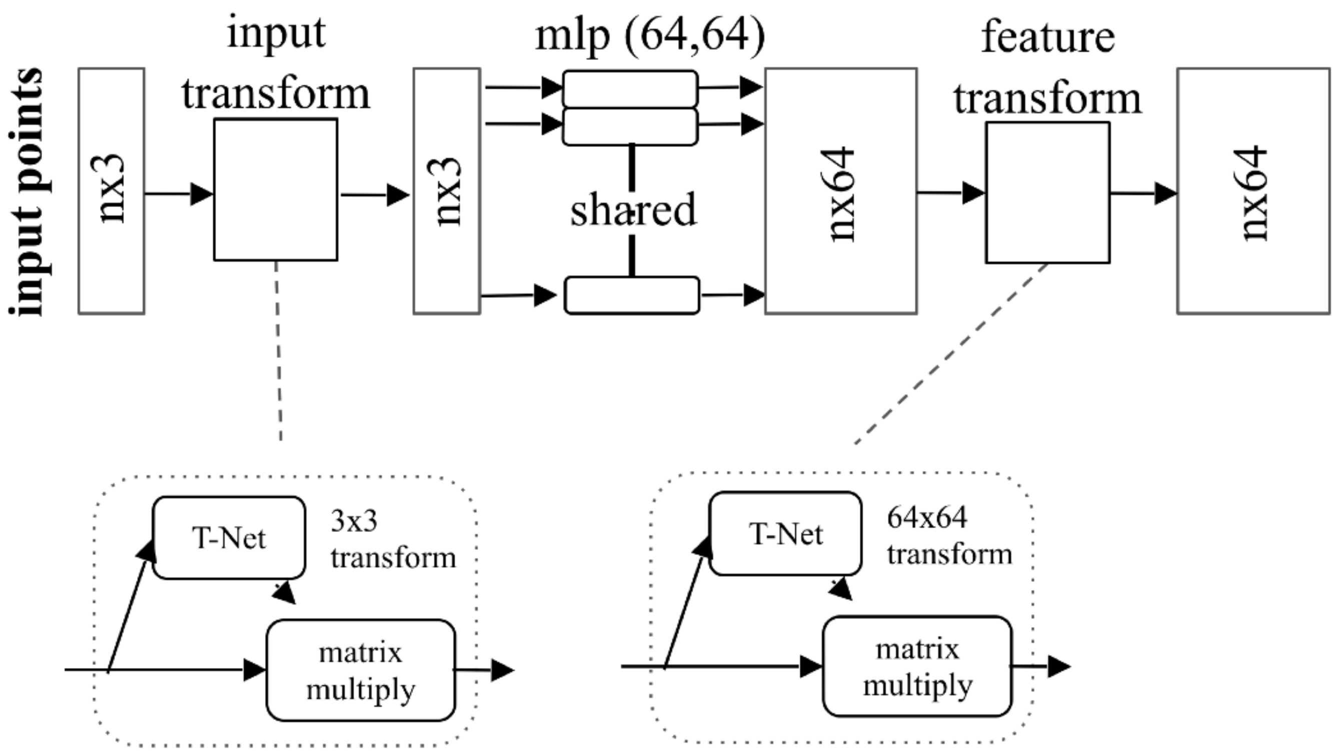
PointNet Classification Network



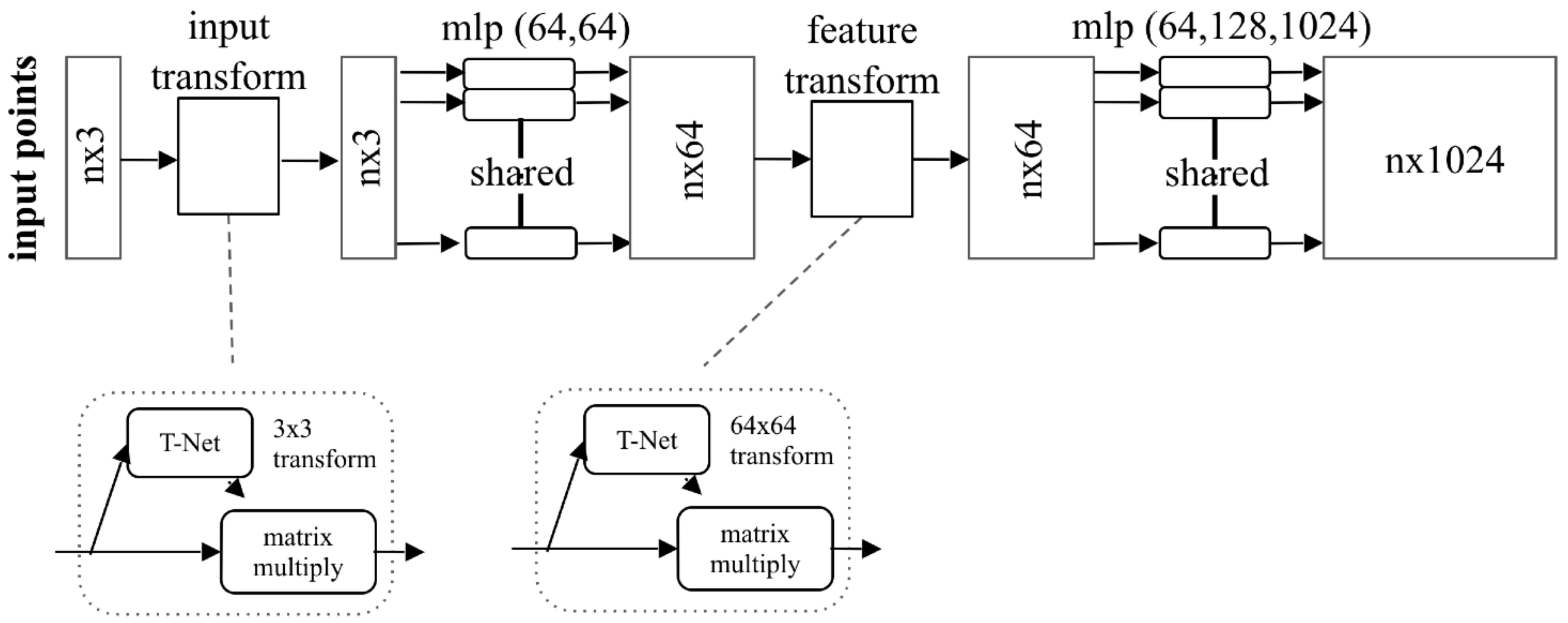
PointNet Classification Network



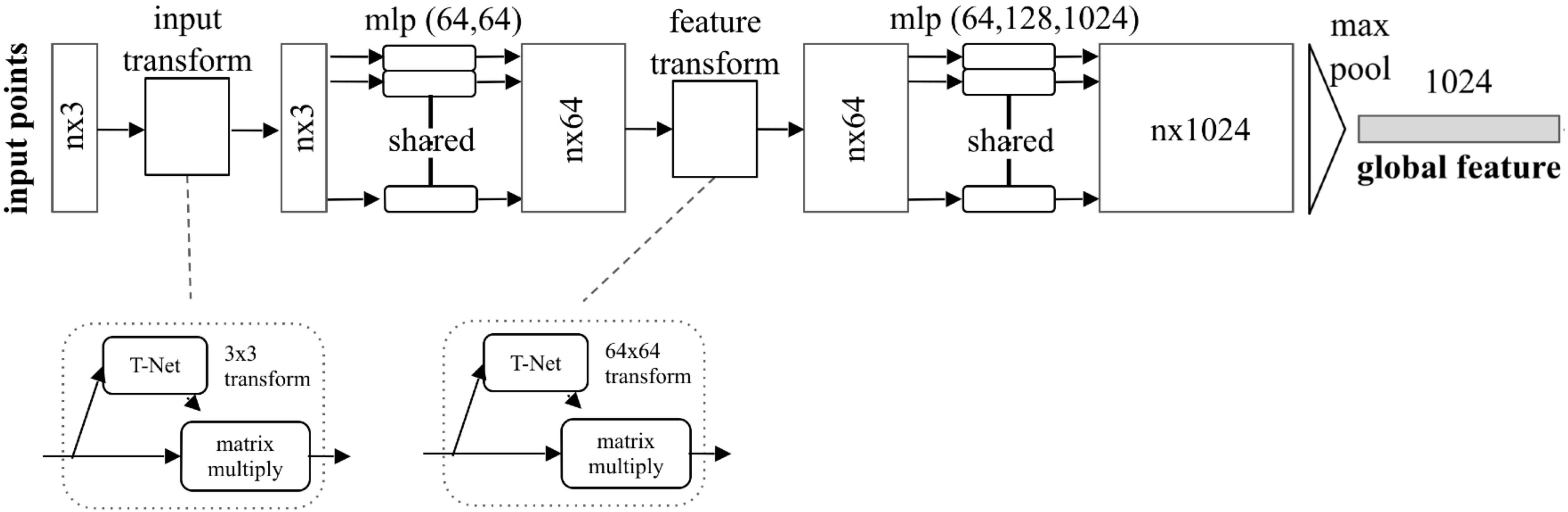
PointNet Classification Network



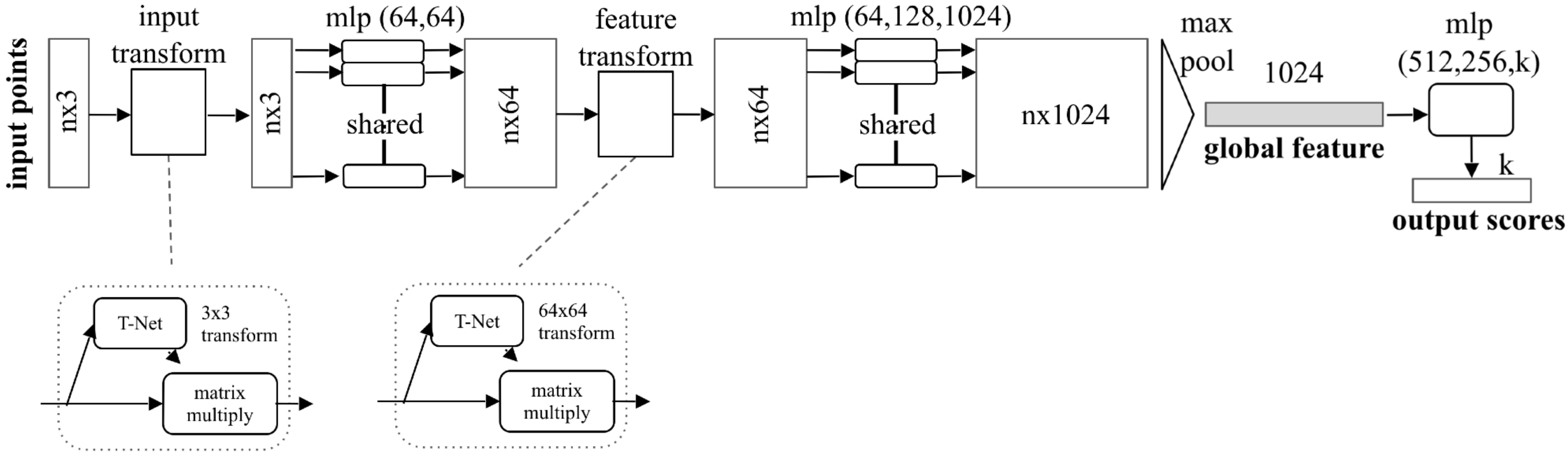
PointNet Classification Network



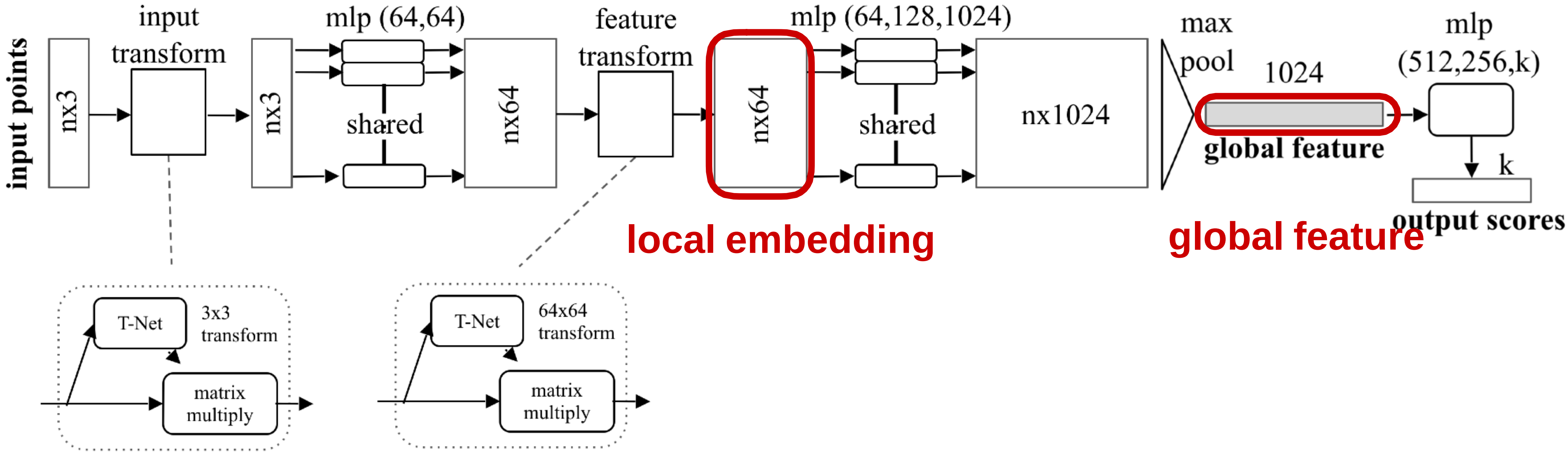
PointNet Classification Network



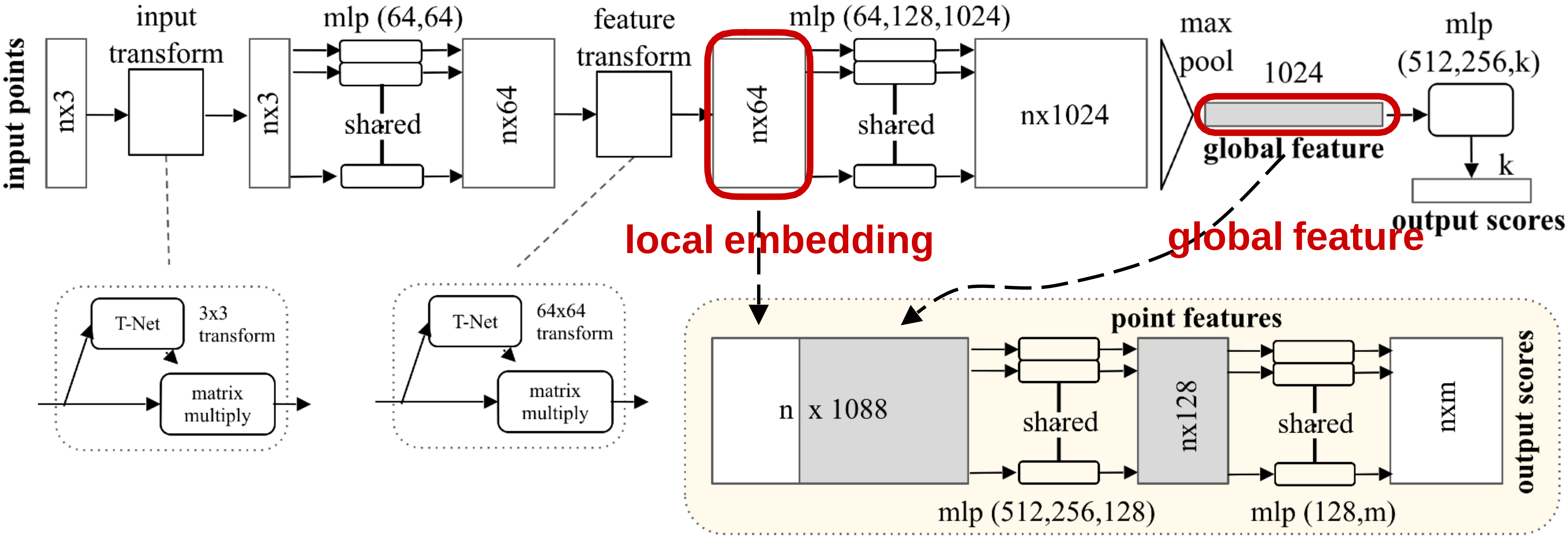
PointNet Classification Network



Extension to PointNet Segmentation Network



Extension to PointNet Segmentation Network



Results

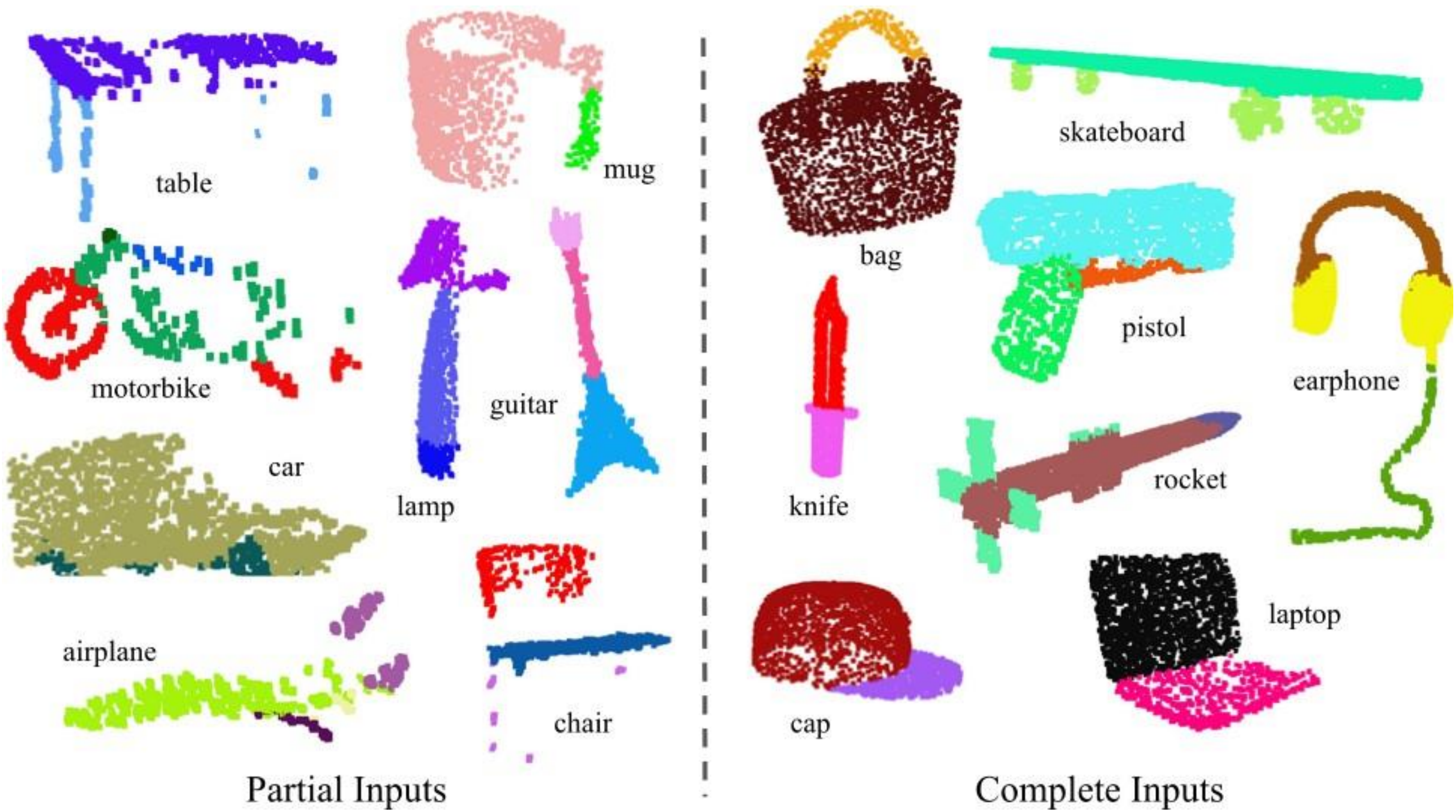
Results on Object Classification

3D CNNs

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

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

Results on Object Part Segmentation



Results on Object Part Segmentation

	mean	aero	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	skate board	table
# shapes		2690	76	55	898	3758	69	787	392	1547	451	202	184	283	66	152	5271
Wu [28]	-	63.2	-	-	-	73.5	-	-	-	74.4	-	-	-	-	-	-	74.8
Yi [30]	81.4	81.0	78.4	77.7	**75.7**	87.6	61.9	**92.0**	85.4	**82.5**	**95.7**	**70.6**	91.9	**85.9**	53.1	69.8	75.3
3DCNN	79.4	75.1	72.8	73.3	70.0	87.2	63.5	88.4	79.6	74.4	93.9	58.7	91.8	76.4	51.2	65.3	77.1
Ours	**83.7**	**83.4**	**78.7**	**82.5**	74.9	**89.6**	**73.0**	91.5	**85.9**	80.8	95.3	65.2	**93.0**	81.2	**57.9**	**72.8**	**80.6**

dataset: ShapeNetPart; metric: mean IoU (%)

Results on Semantic Scene Parsing

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

Robustness to Data Corruption

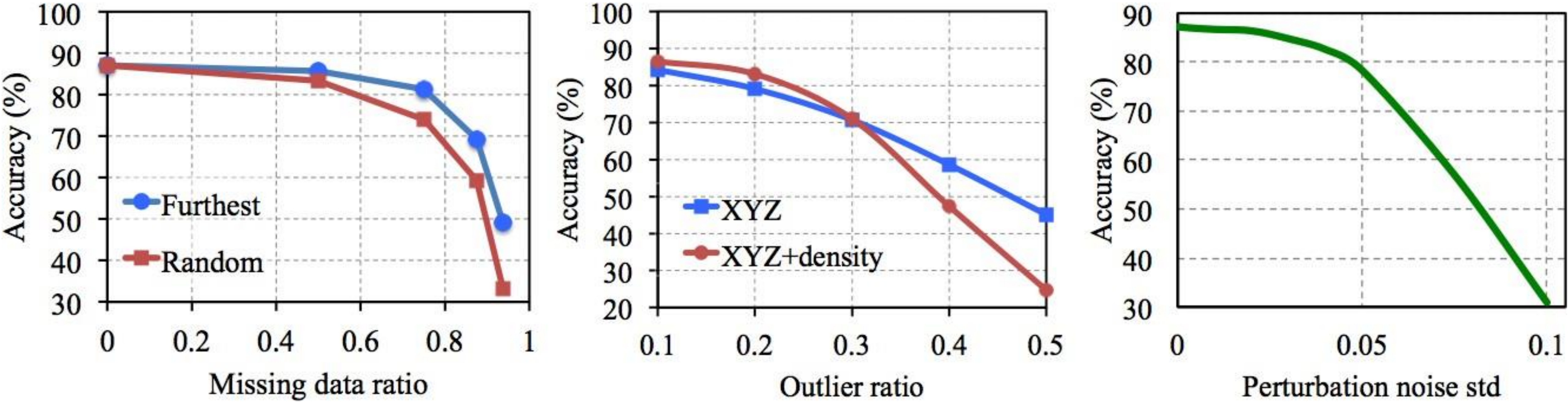
dataset: ModelNet40; metric: 40-class classification accuracy (%)

Robustness to Data Corruption

Less than 2% accuracy drop with 50% missing data

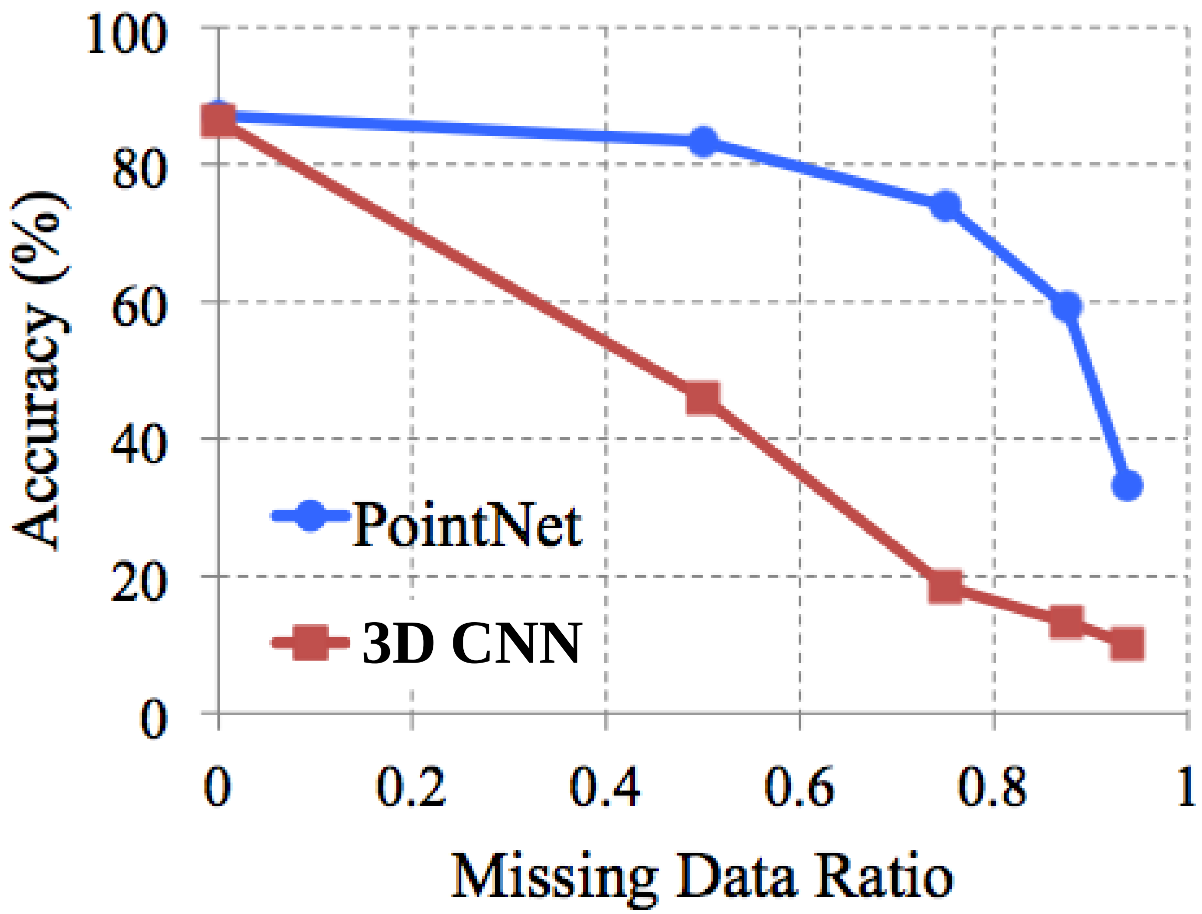
dataset: ModelNet40; metric: 40-class classification accuracy (%)

Robustness to Data Corruption



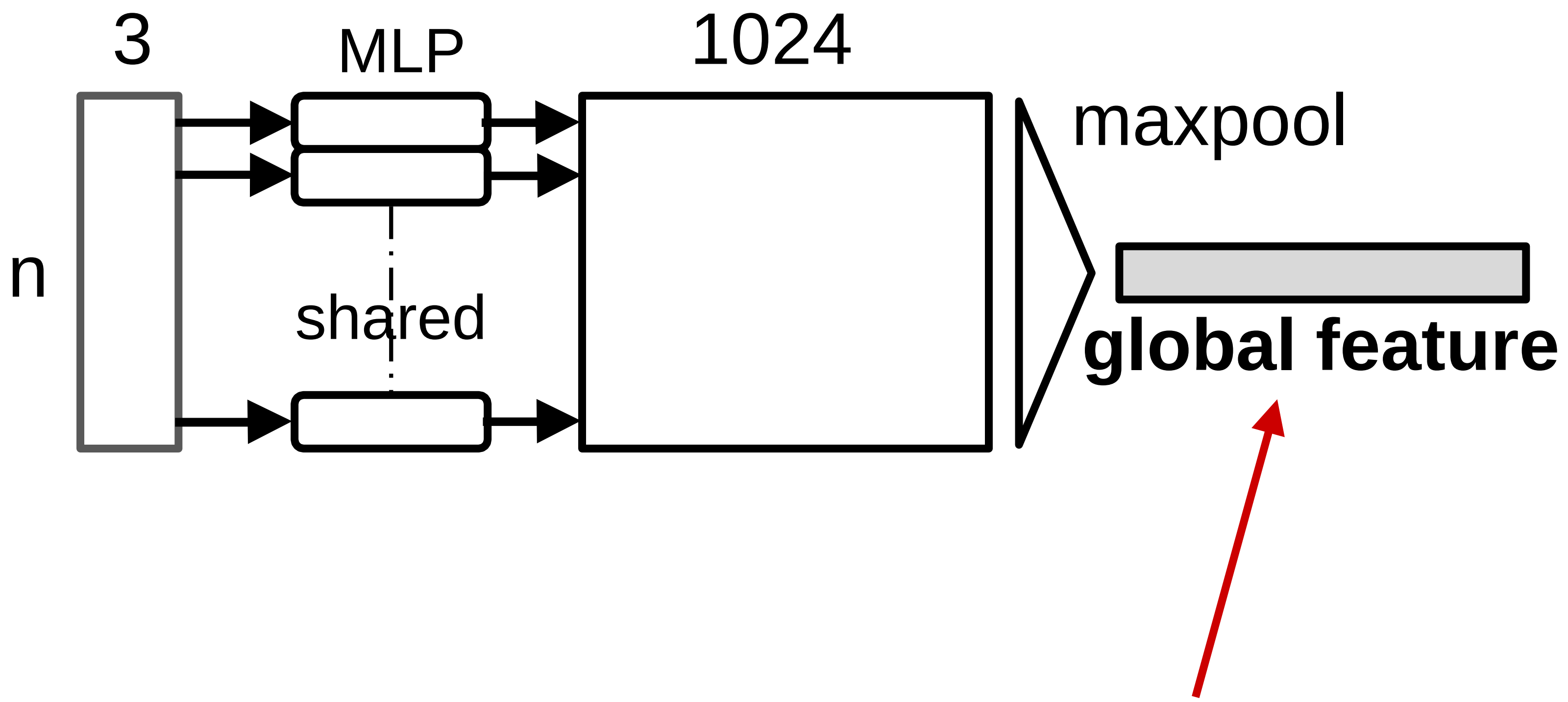
dataset: ModelNet40; metric: 40-class classification accuracy (%)

Robustness to Data Corruption



Why is PointNet so robust to missing data?

Visualizing Global Point Cloud Features



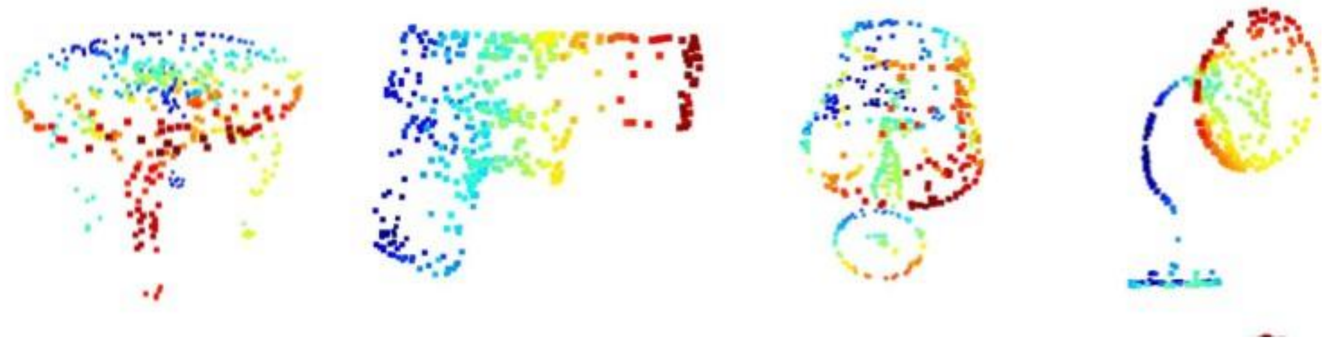
<u>Which input points</u> are contributing to the global feature?

(critical points)

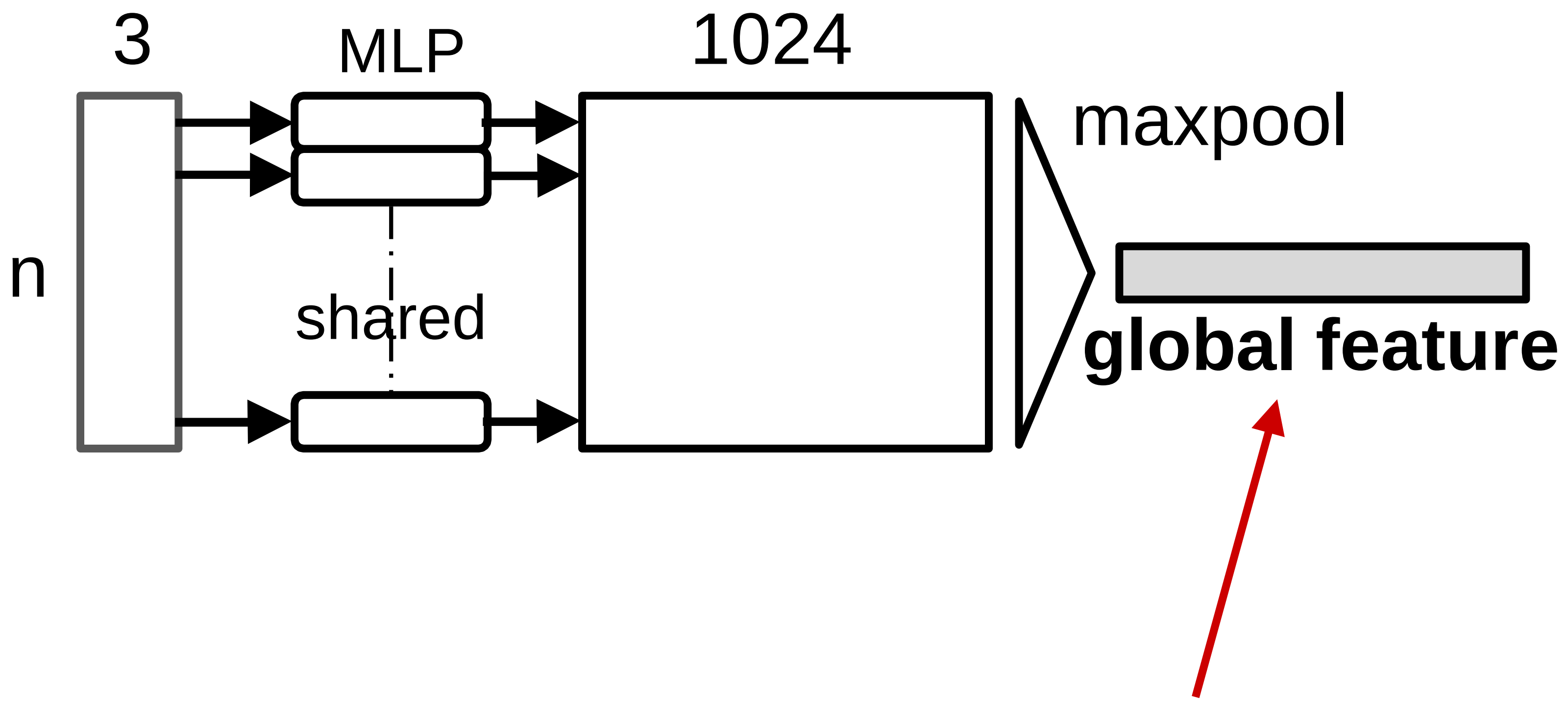
Visualizing Global Point Cloud Features

Original Shape:

Critical Point Sets:



Visualizing Global Point Cloud Features



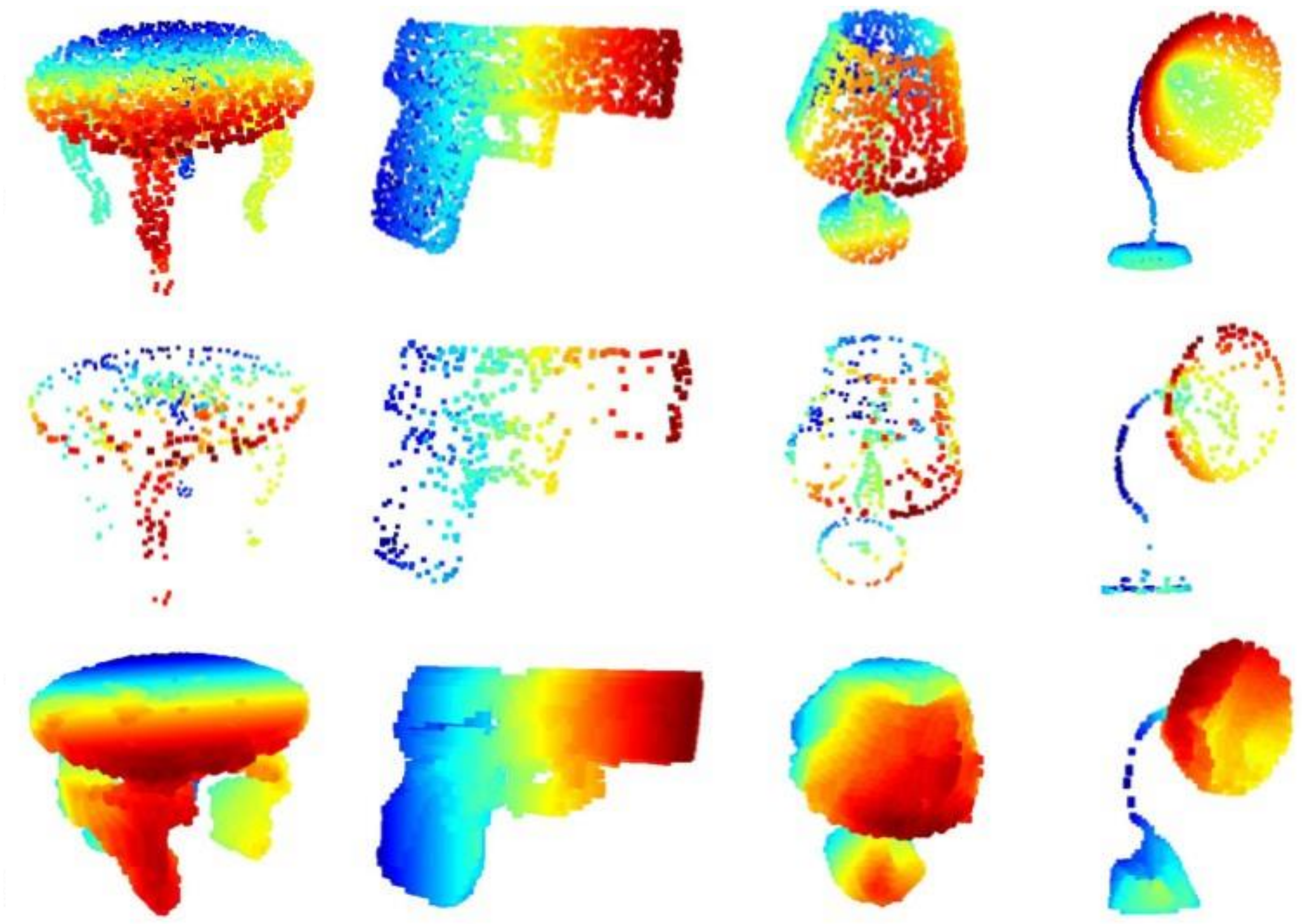
Which points won't affect the global feature?

Visualizing Global Point Cloud Features

Original Shape:

Critical Point Set:

Upper bound set:

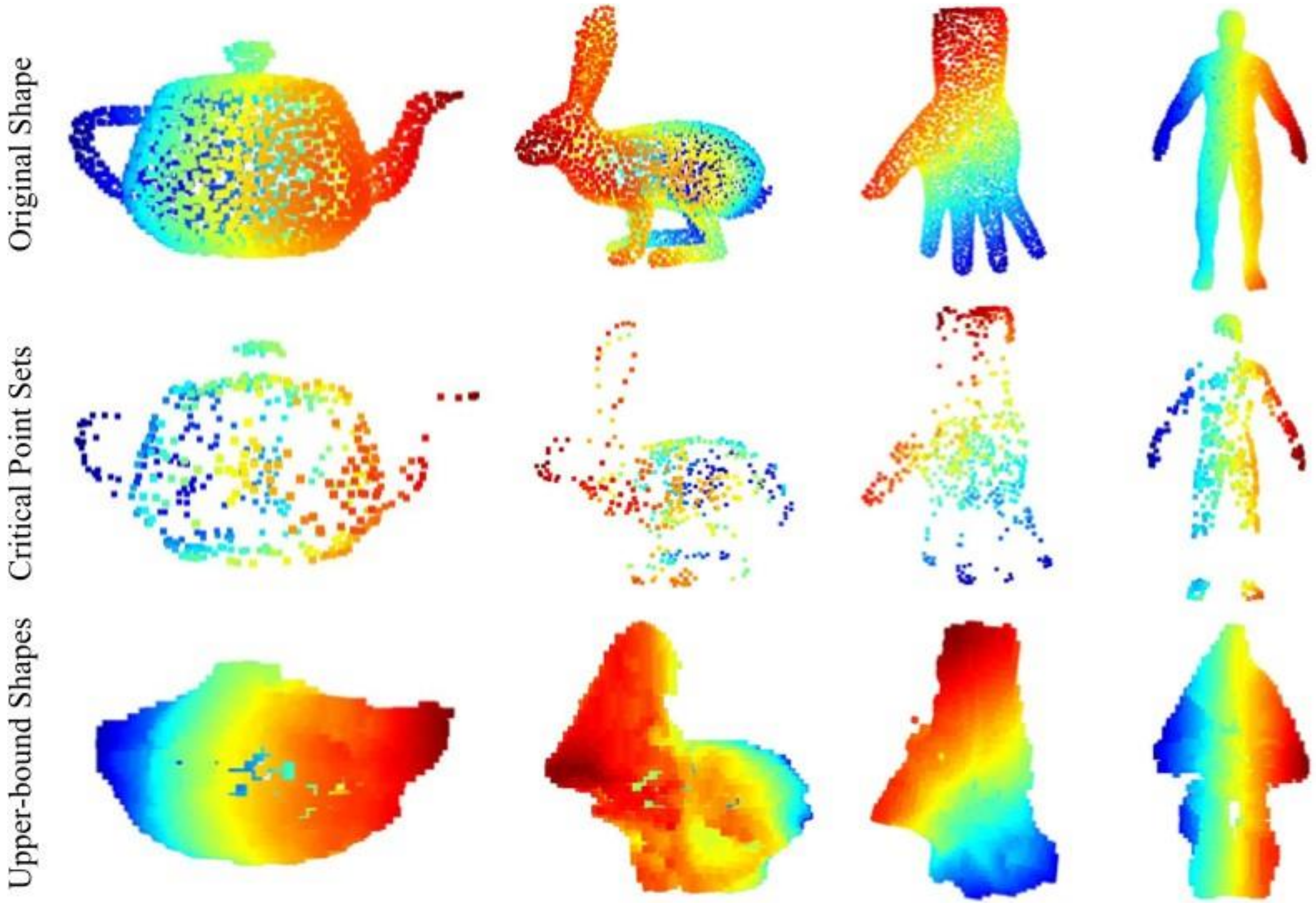


Visualizing Global Point Cloud Features (OOS)

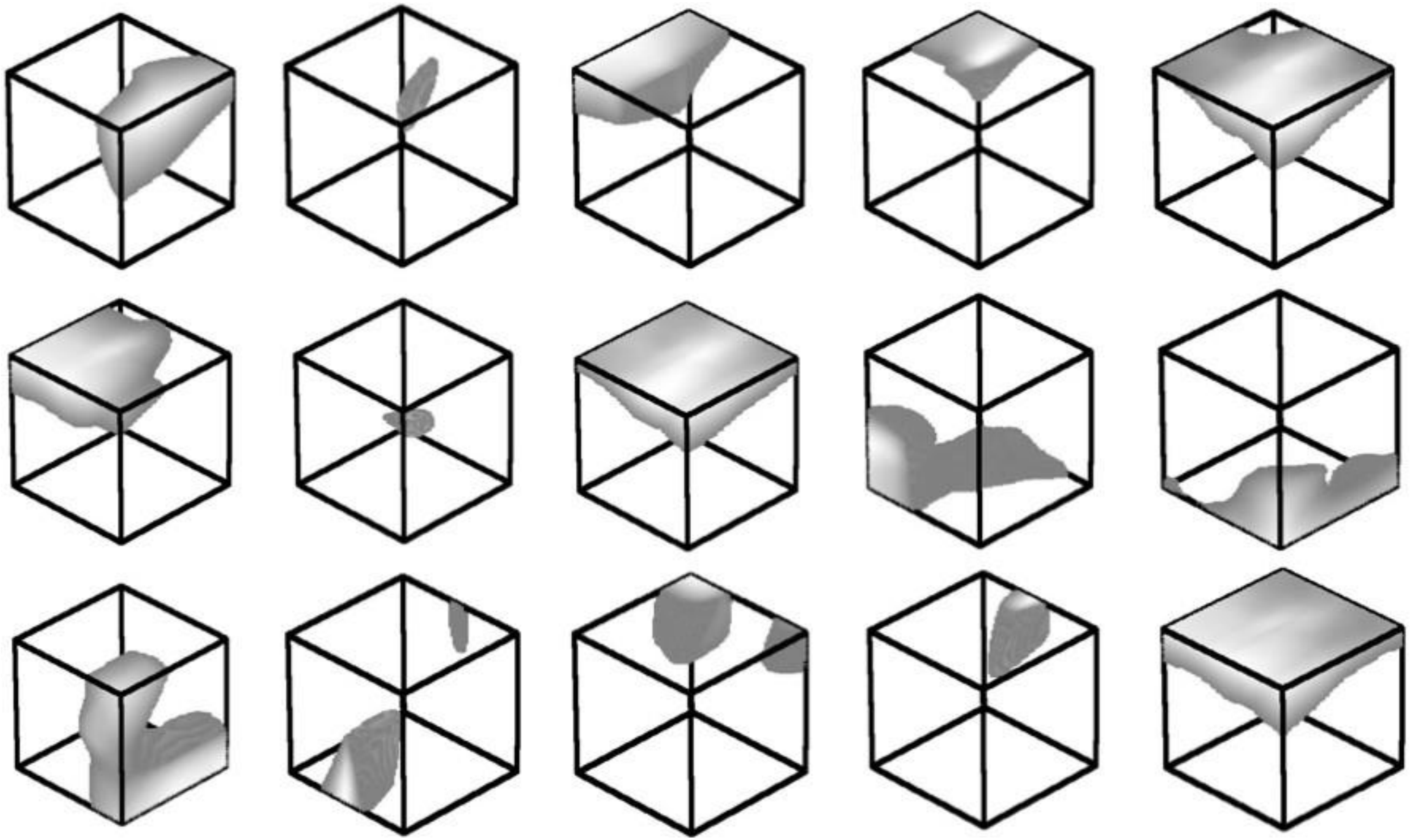
Original Shape:

Critical Point Set:

Upper bound Set:

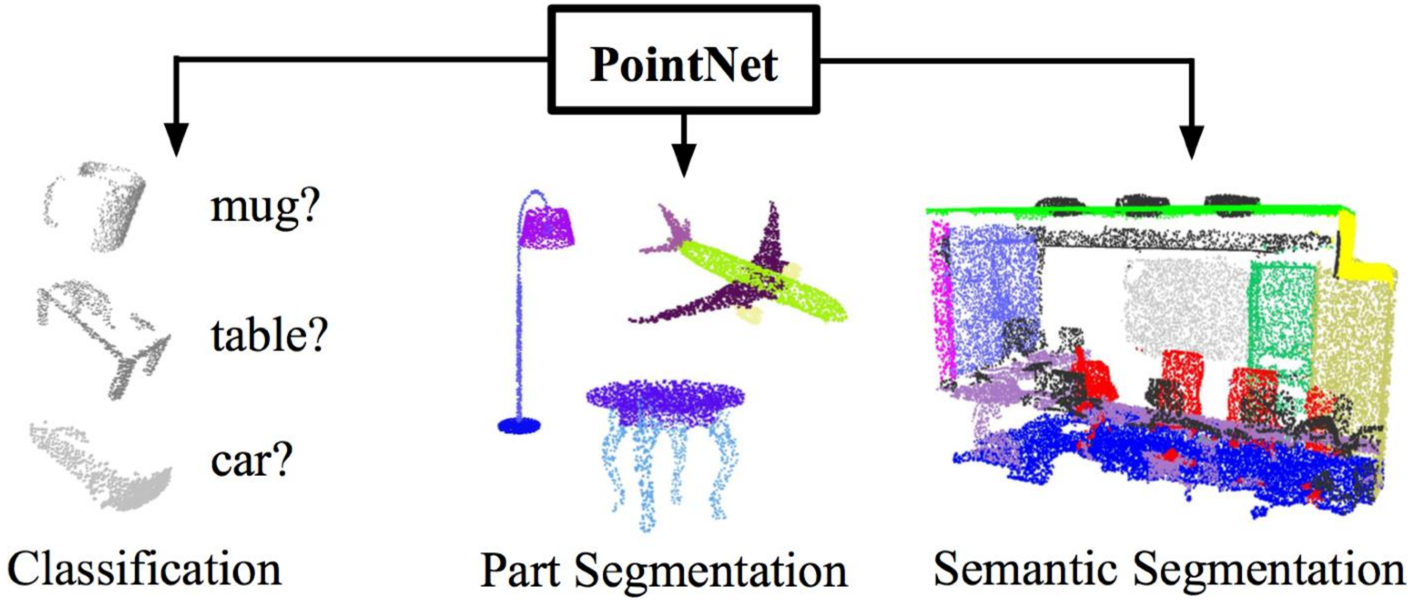


Visualizing Point Functions



Conclusion

- PointNet is a novel deep neural network that directly consumes point cloud.
- A unified approach to various 3D recognition tasks.
- Rich theoretical analysis and experimental results.



Code & Data Available!
http://stanford.edu/~rqi/pointnet

Speed and Model Size

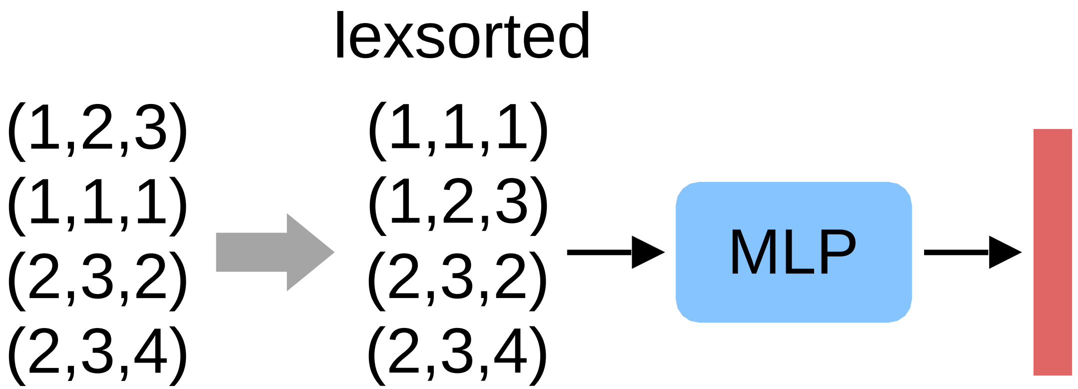
	#params	FLOPs/sample
PointNet (vanilla)	0.8M	148M
PointNet	3.5M	440M
Subvolume [16]	16.6M	3633M
MVCNN [20]	60.0M	62057M

Inference time 11.6ms, 25.3ms GTX1080, batch size 8

Permutation Invariance: How about Sorting?

“Sort” the points before feeding them into a network.

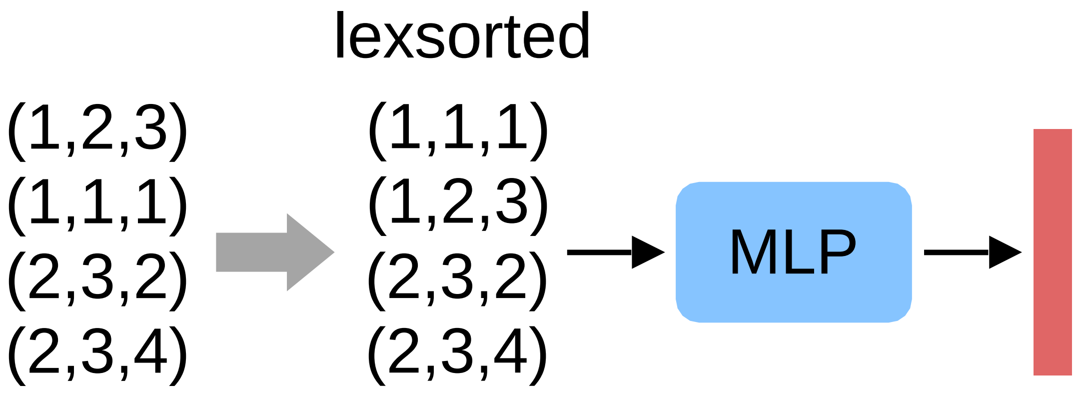
Unfortunately, there is no canonical order in high dim space.



Permutation Invariance: How about Sorting?

"Sort" the points before feeding them into a network.

Unfortunately, there is no canonical order in high dim space.



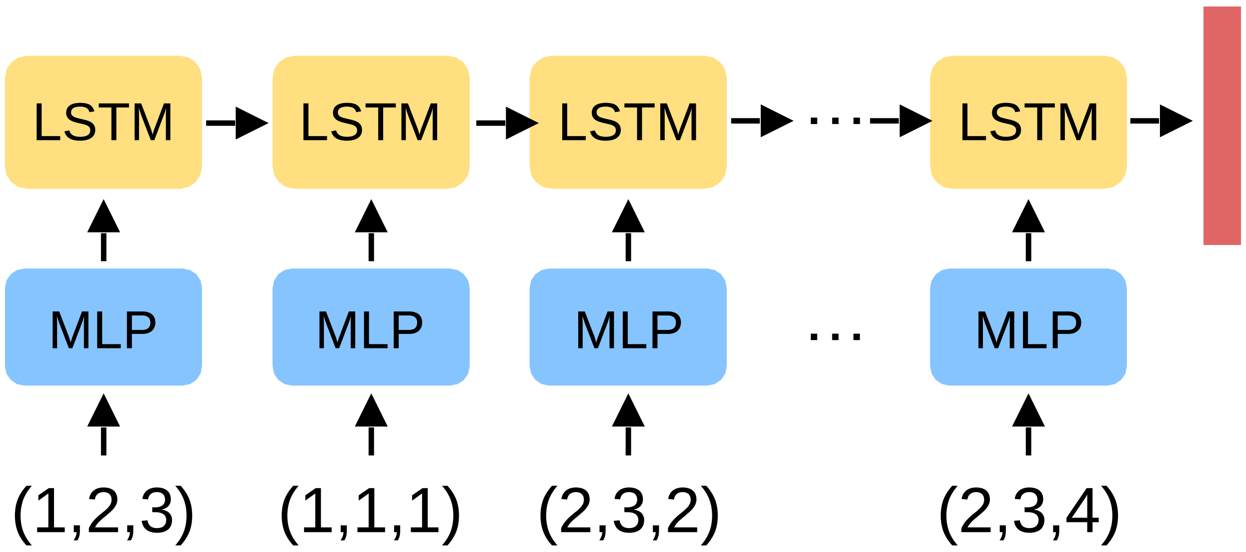
Multi-Layer Perceptron
(ModelNet shape classification)

	Accuracy
Unordered Input	12%
Lexsorted Input	40%
PointNet (vanilla)	**87%**

Permutation Invariance: How about RNNs?

Train RNN with permutation augmentation.

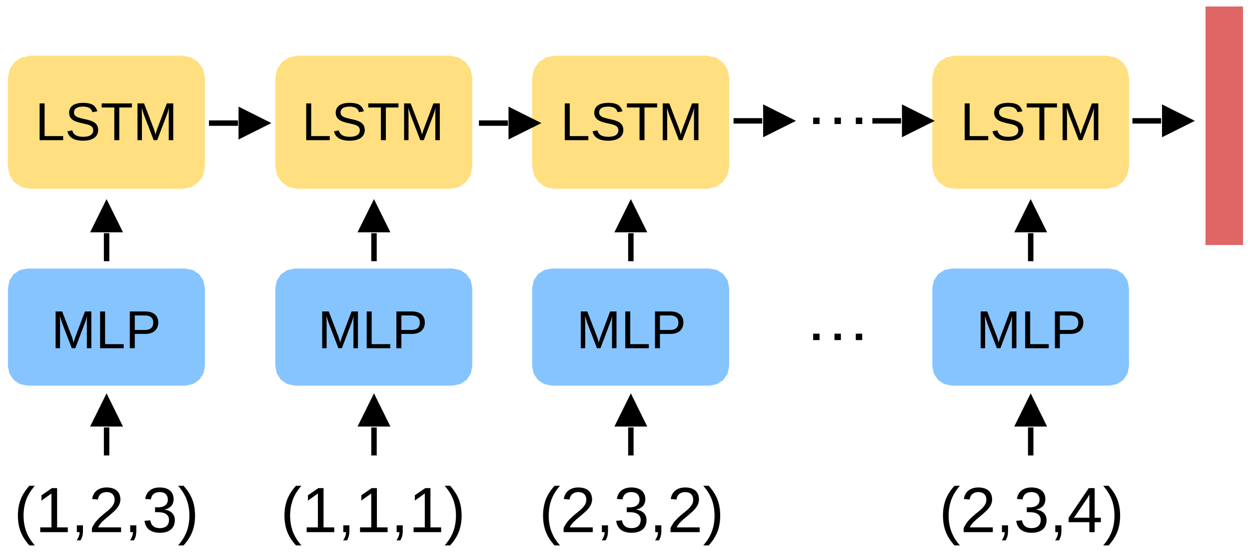
However, RNN forgets and order matters.



Permutation Invariance: How about RNNs?

Train RNN with permutation augmentation.

However, RNN forgets and order matters.



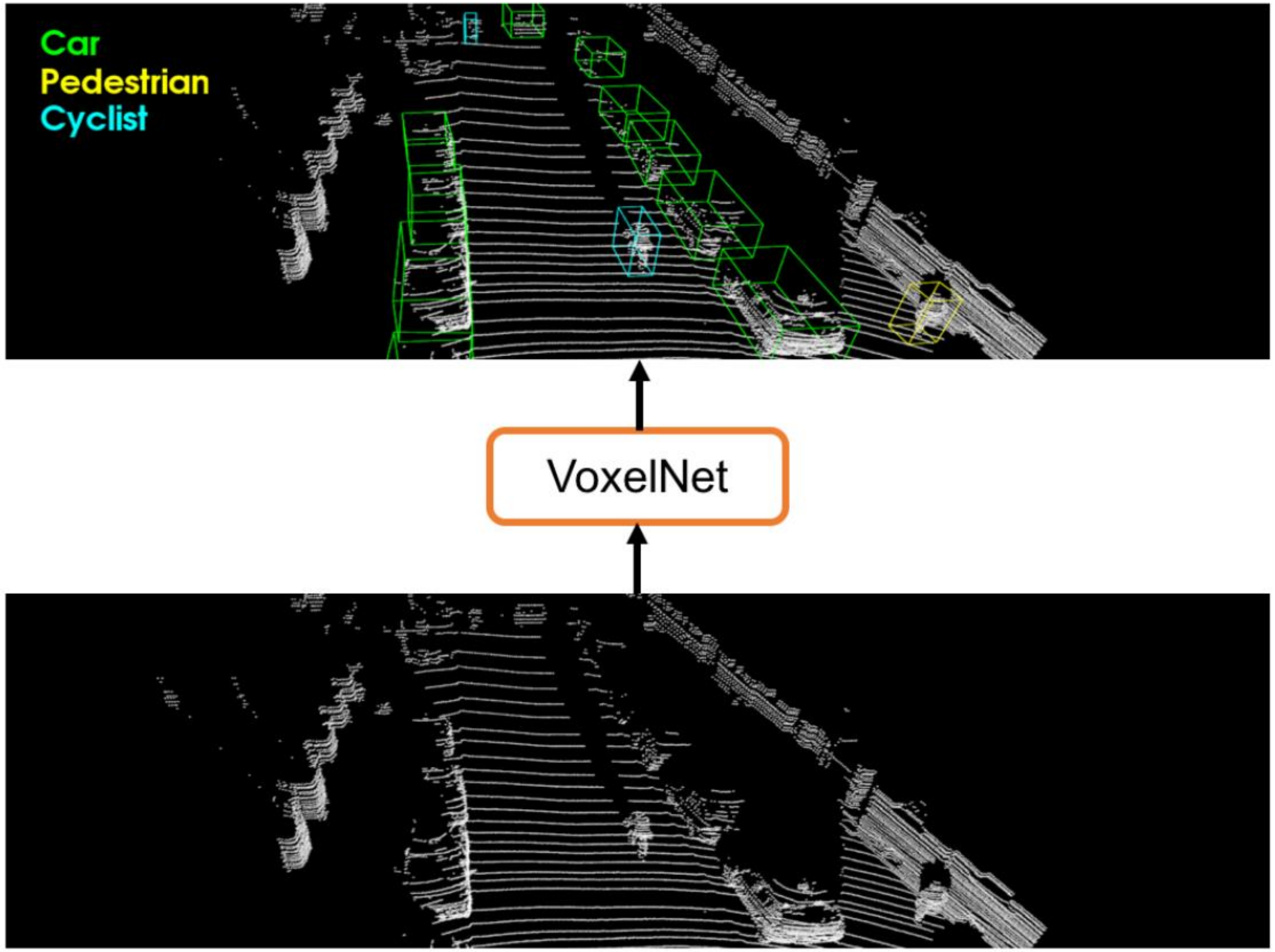
LSTM Network
(ModelNet shape classification)

	Accuracy
LSTM	75%
PointNet (vanilla)	**87%**

Outline

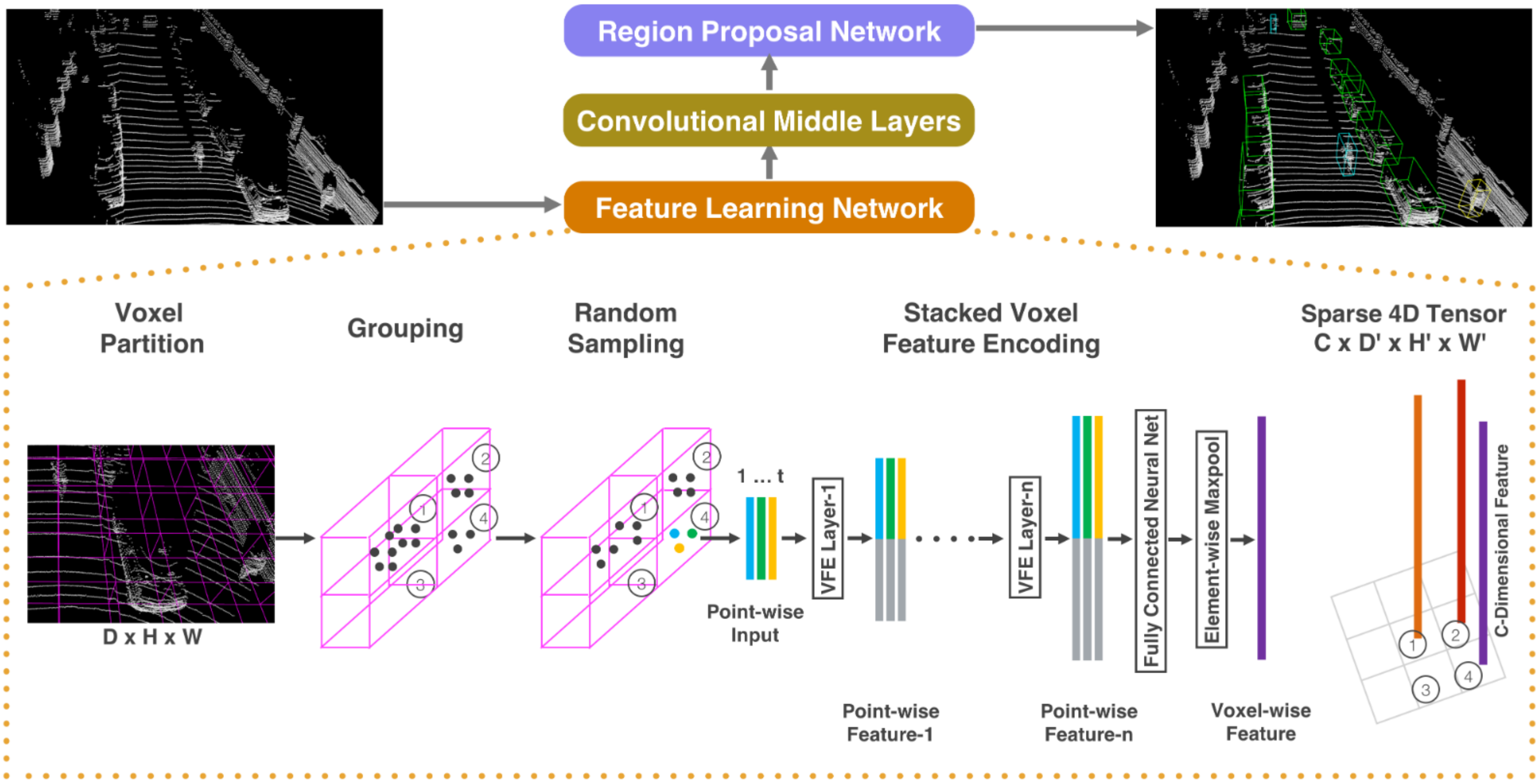
- What is lidar?
- How do we make decisions about point clouds?
  - PointNet – orderless point processing
  - VoxelNet – voxel-based point processing
  - PointPillars – bird's eye view point processing
    - Exploiting Visibility for 3D Object Detection
  - LaserNet – range image point processing
- PseudoLidar – Bird's eye view depth map processing

VoxelNet

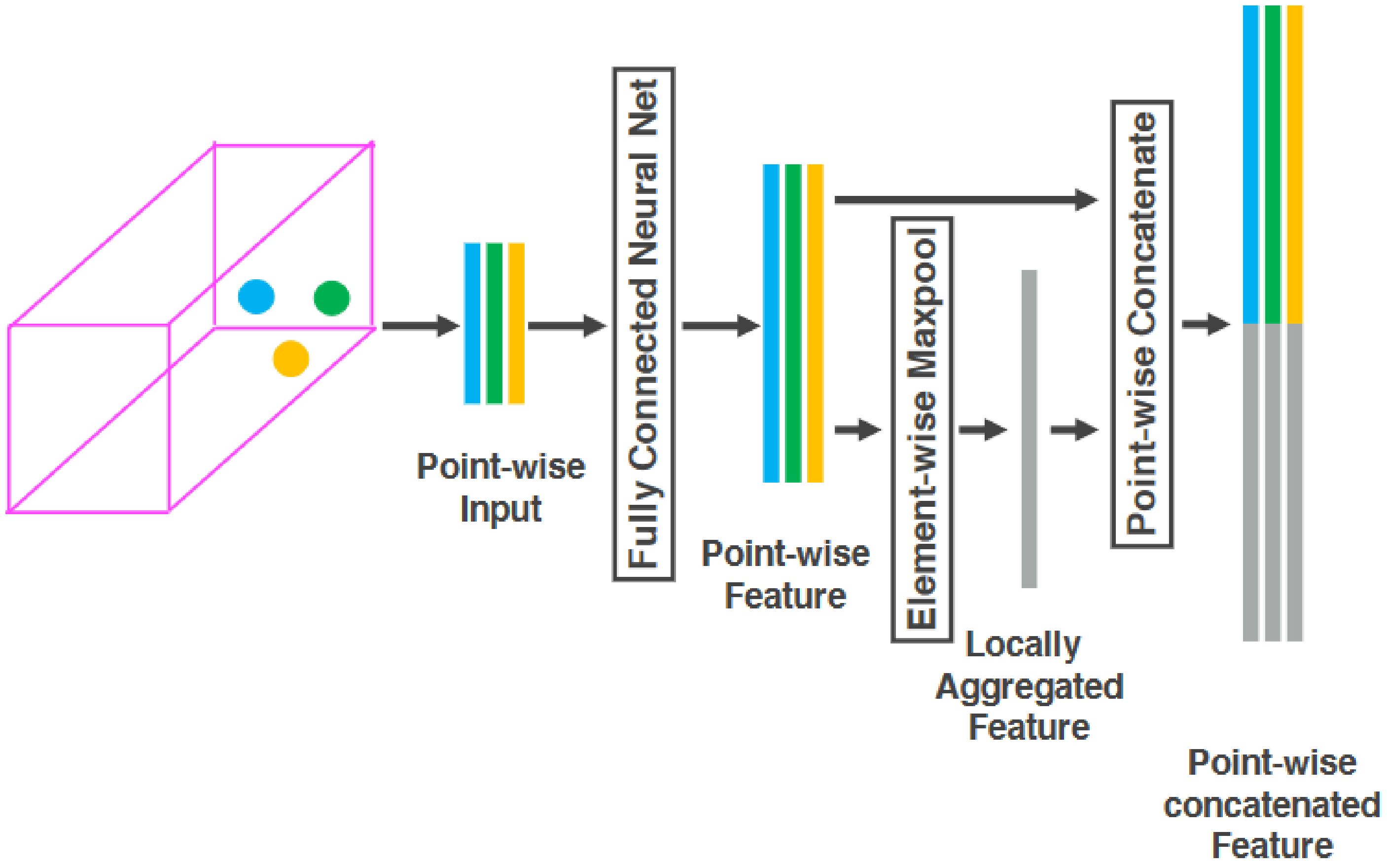


VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection
Yin Zhou and Oncel Tuzel. CVPR 2018

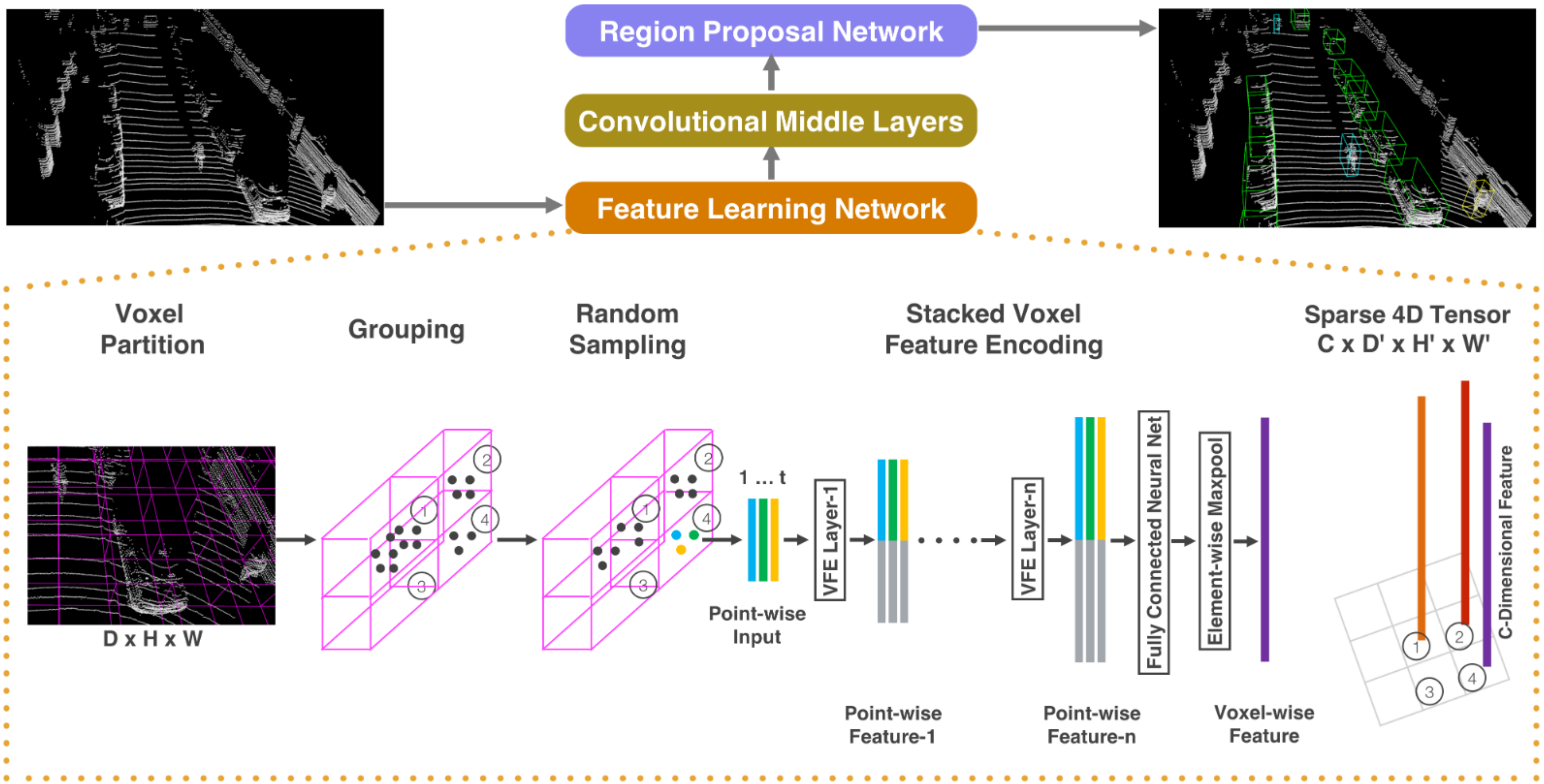
VoxelNet Overview



VoxelNet Voxel encoding looks a lot like PointNet



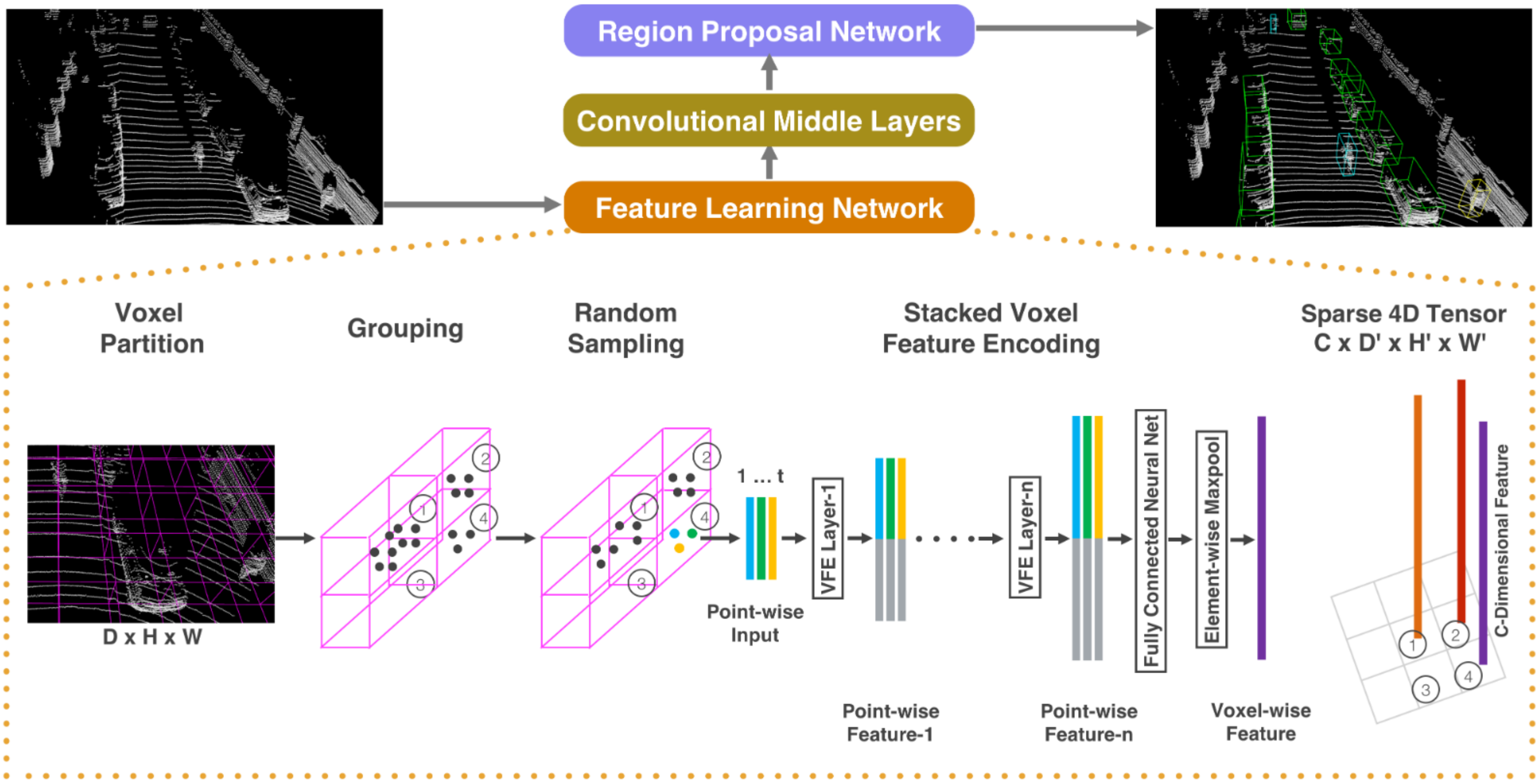
VoxelNet Overview



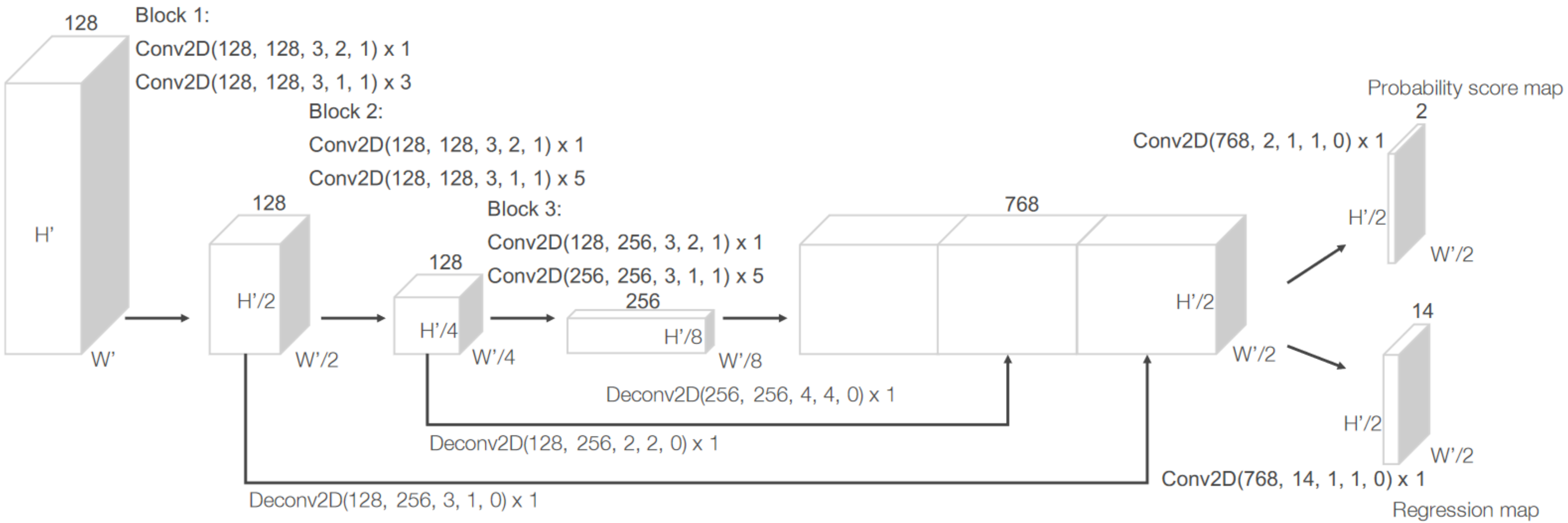
VoxelNet “Convolutional Middle Layers”

- For car detection, divide the world into 10 x 400 x 352 voxels, corresponding to voxels that are 40 cm tall and 20 cm in width/length.
- Uses **3D** convolutions instead of 2D as we’ve seen before.
- The Z / height dimension gets downsampled away after many layers

VoxelNet Overview

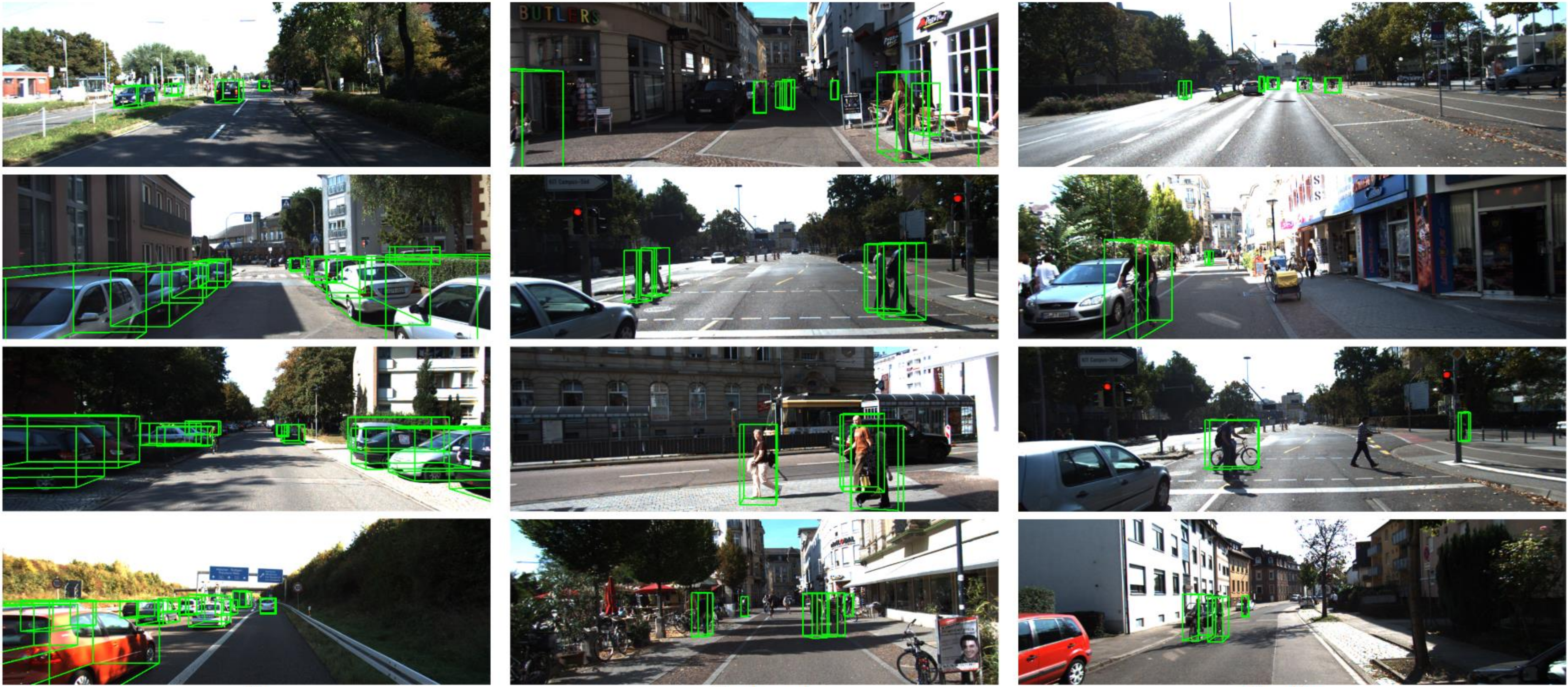


VoxelNet Region Proposal Network



$$(x_c^g, y_c^g, z_c^g, l^g, w^g, h^g, \theta^g)$$

VoxelNet qualitative results



Car Pedestrian Cyclist

VoxelNet quantitative results

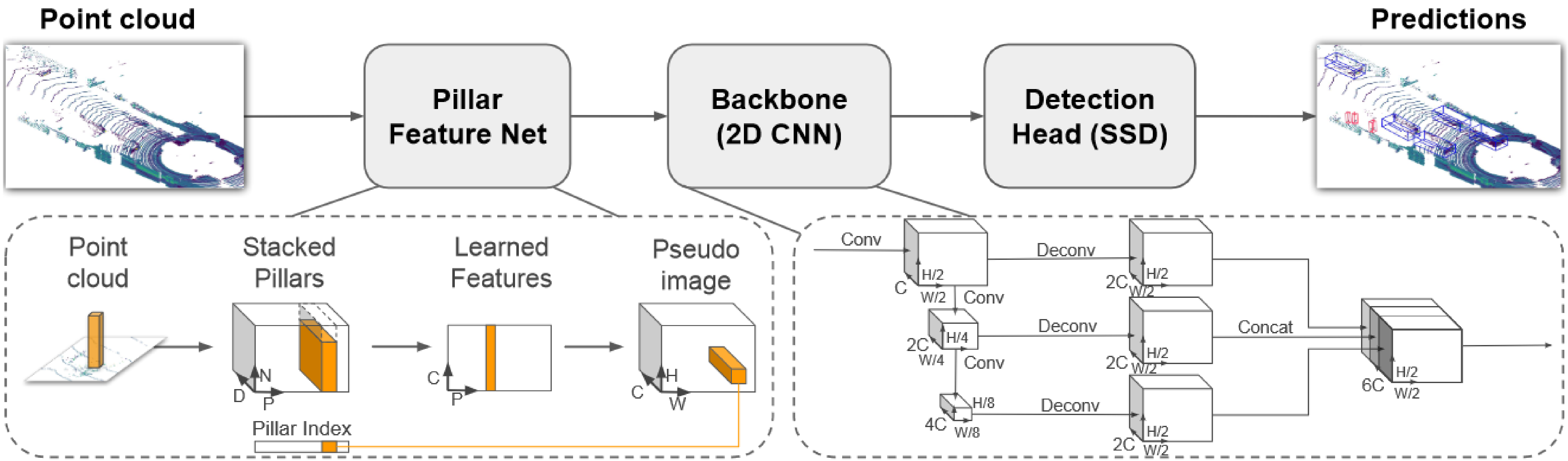
Method	Modality	Car			Pedestrian			Cyclist		
		Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Mono3D [3]	Mono	2.53	2.31	2.31	N/A	N/A	N/A	N/A	N/A	N/A
3DOP [4]	Stereo	6.55	5.07	4.10	N/A	N/A	N/A	N/A	N/A	N/A
VeloFCN [22]	LiDAR	15.20	13.66	15.98	N/A	N/A	N/A	N/A	N/A	N/A
MV (BV+FV) [5]	LiDAR	71.19	56.60	55.30	N/A	N/A	N/A	N/A	N/A	N/A
MV (BV+FV+RGB) [5]	LiDAR+Mono	71.29	62.68	56.56	N/A	N/A	N/A	N/A	N/A	N/A
HC-baseline	LiDAR	71.73	59.75	55.69	43.95	40.18	37.48	55.35	36.07	34.15
VoxelNet	LiDAR	**81.97**	**65.46**	**62.85**	**57.86**	**53.42**	**48.87**	**67.17**	**47.65**	**45.11**

Evaluation on KITTI according to 3D IoU

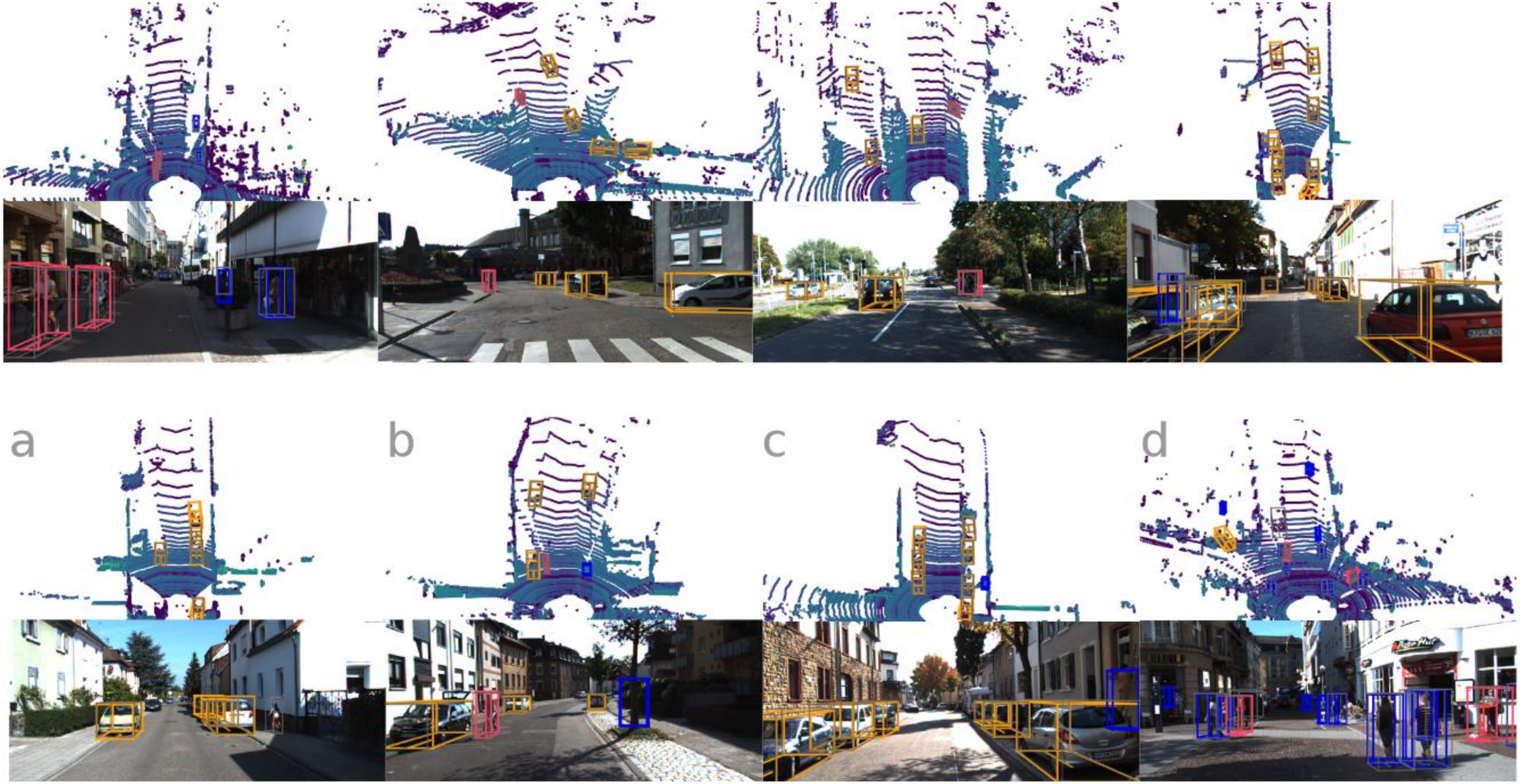
Outline

- What is lidar?
- How do we make decisions about point clouds?
    - PointNet – orderless point processing
    - VoxelNet – voxel-based point processing
    - PointPillars – bird’s eye view point processing
        - Exploiting Visibility for 3D Object Detection
    - LaserNet – range image point processing
- PseudoLidar – Bird’s eye view depth map processing

PointPillars



PointPillars: Fast Encoders for Object Detection from Point Clouds
Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, Oscar Beijbom. CVPR 2019


a
b
c
d

Outline

- What is lidar?
- How do we make decisions about point clouds?
  - PointNet – orderless point processing
  - VoxelNet – voxel-based point processing
  - PointPillars – bird’s eye view point processing
    - Exploiting Visibility for 3D Object Detection
  - LaserNet – range image point processing
- PseudoLidar – Bird’s eye view depth map processing

What You See Is What You Get
Exploiting Visibility for 3D Object Detection

Peiyun Hu, Jason Ziglar, David Held, Deva Ramanan

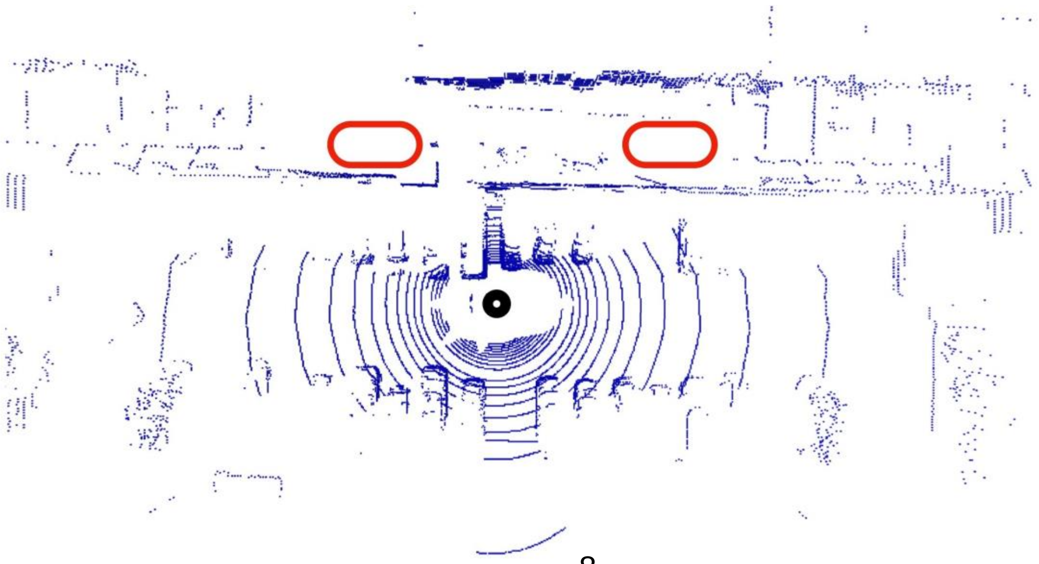
Carnegie Mellon University Argo AI

THE ROBOTICS INSTITUTE

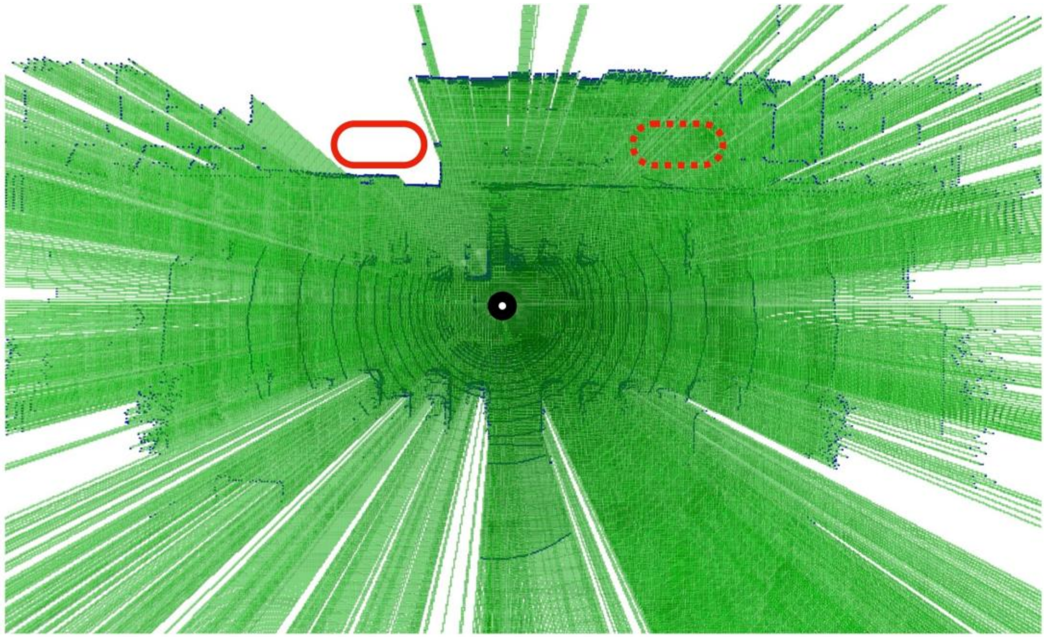
CVPR 2020

ARGO AI

What is a good representation for LiDAR data?

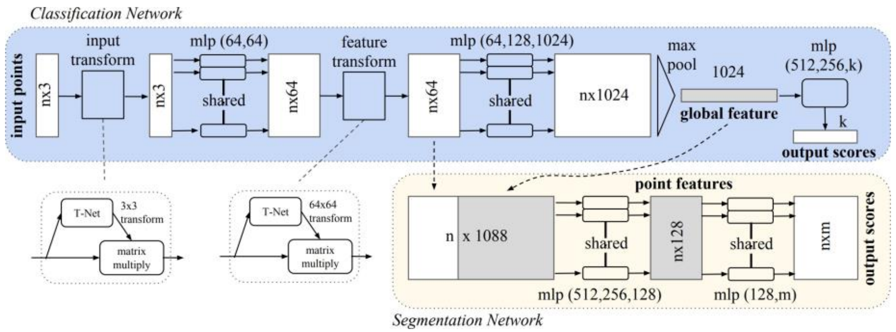


8
8



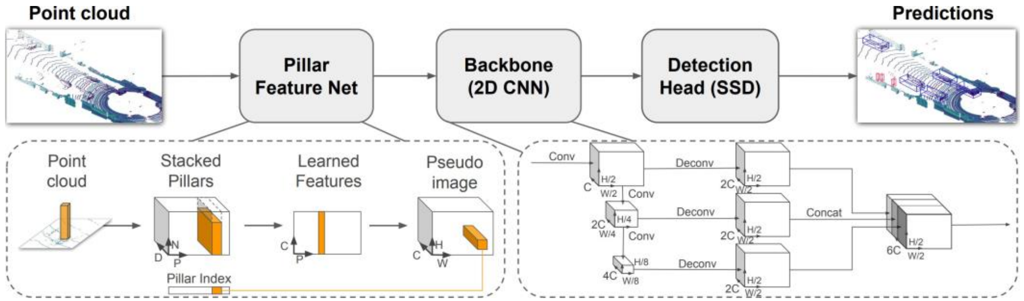
- LiDAR data provides more than just point measurements
- Rays emanating from the sensor to each 3D point must pass through free space
- Representing LiDAR data as (x, y, z)s fundamentally destroys such freespace information

What representations do we have?



Deep Point Representation

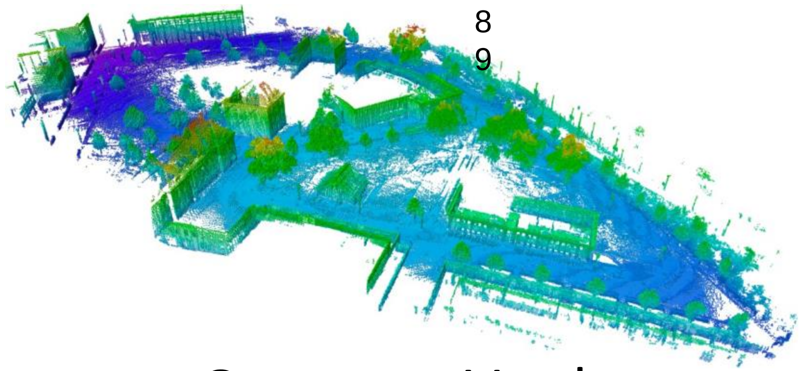
PointNet, Qi et al., CVPR'17



Deep Voxel Representation

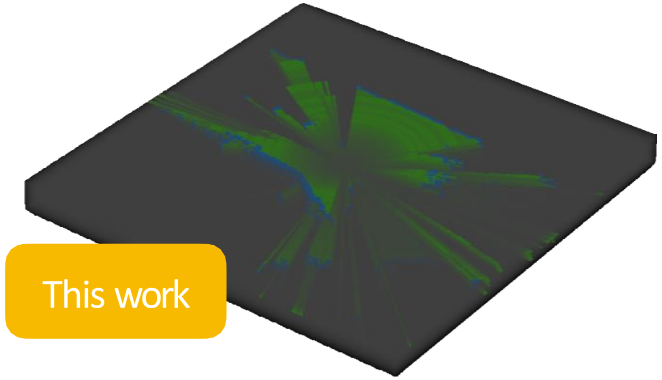
PointPillars, Lang et al., CVPR'19

8
9



Occupancy Voxels

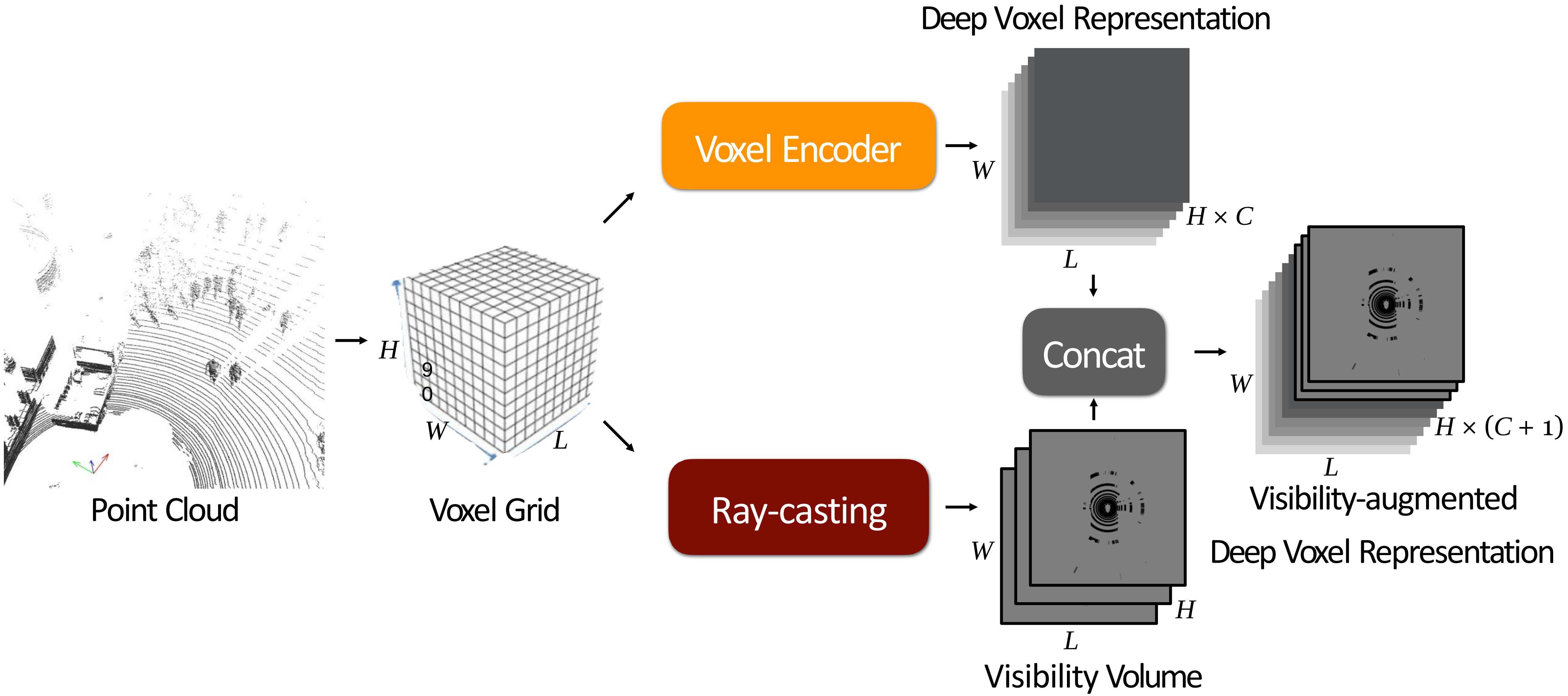
OctoMap, Hornung et al., Autonomous Robots'13



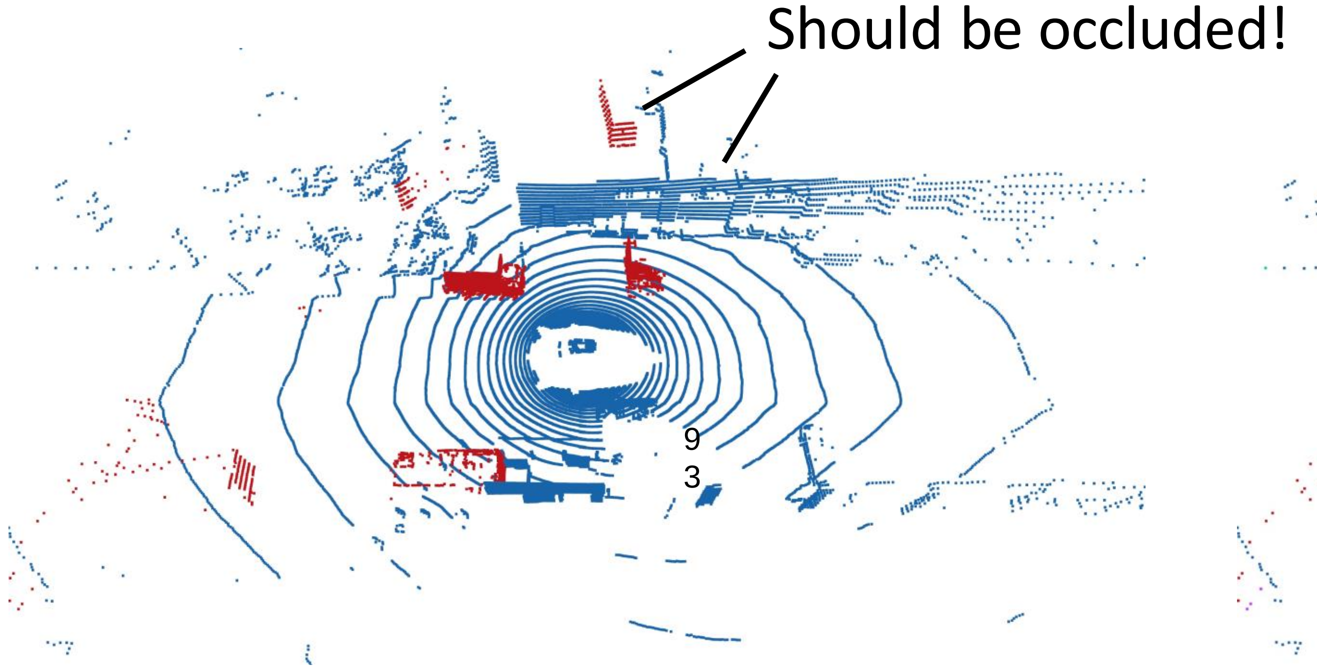
Visibility Augmented Deep Voxels

WYSIWYG, Hu et al., CVPR'20

A Simple Approach to Augment Visibility



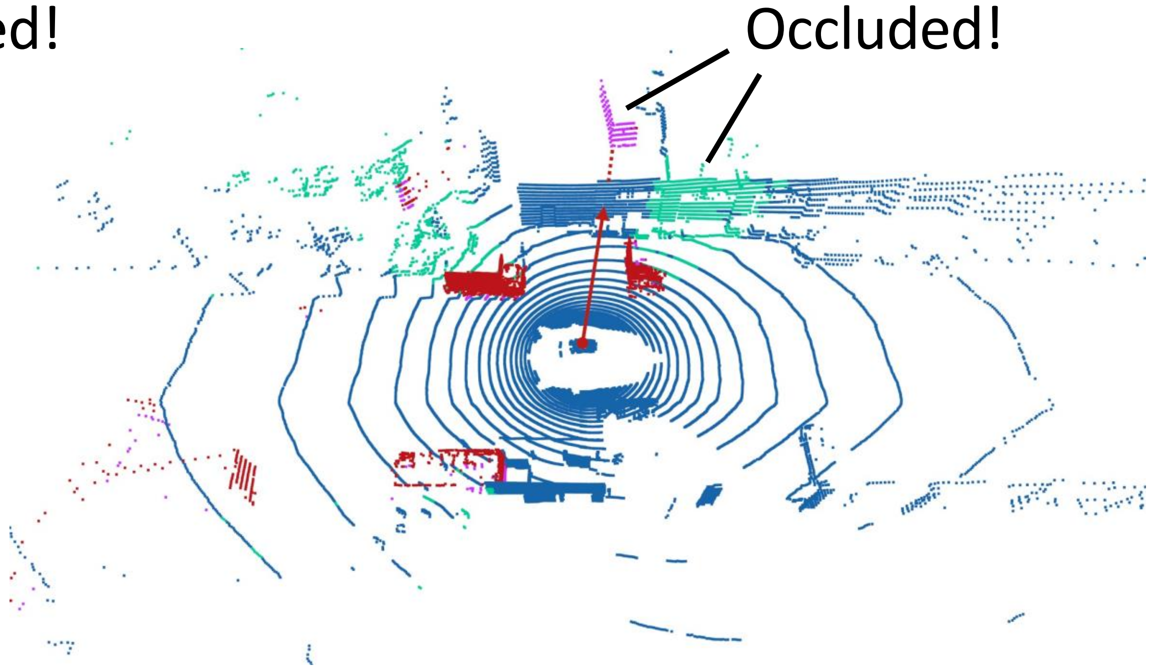
Visibility-aware LiDAR Synthesis



Naive Object Augmentation

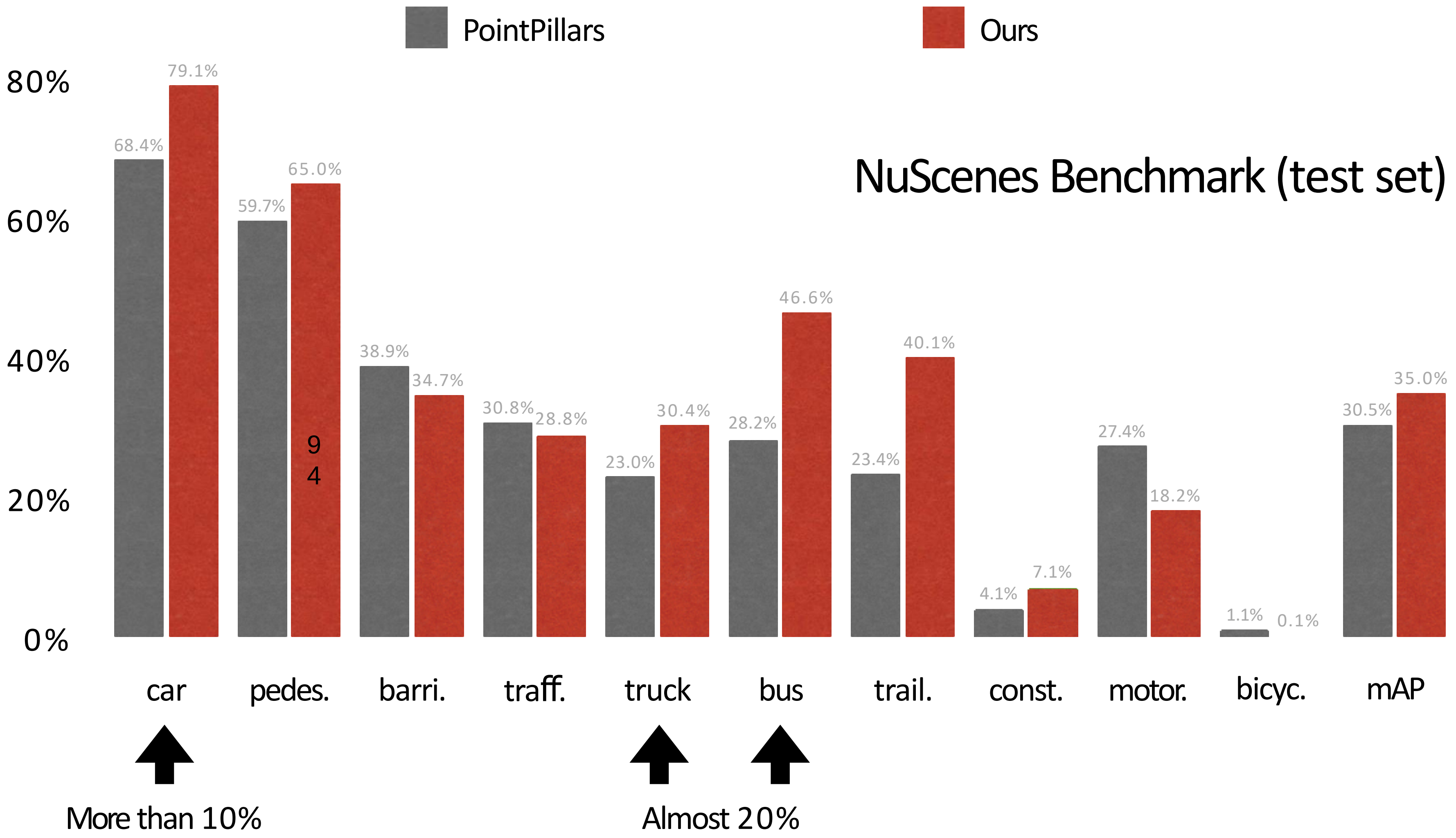
PointPillars, Lang et al., CVPR'19

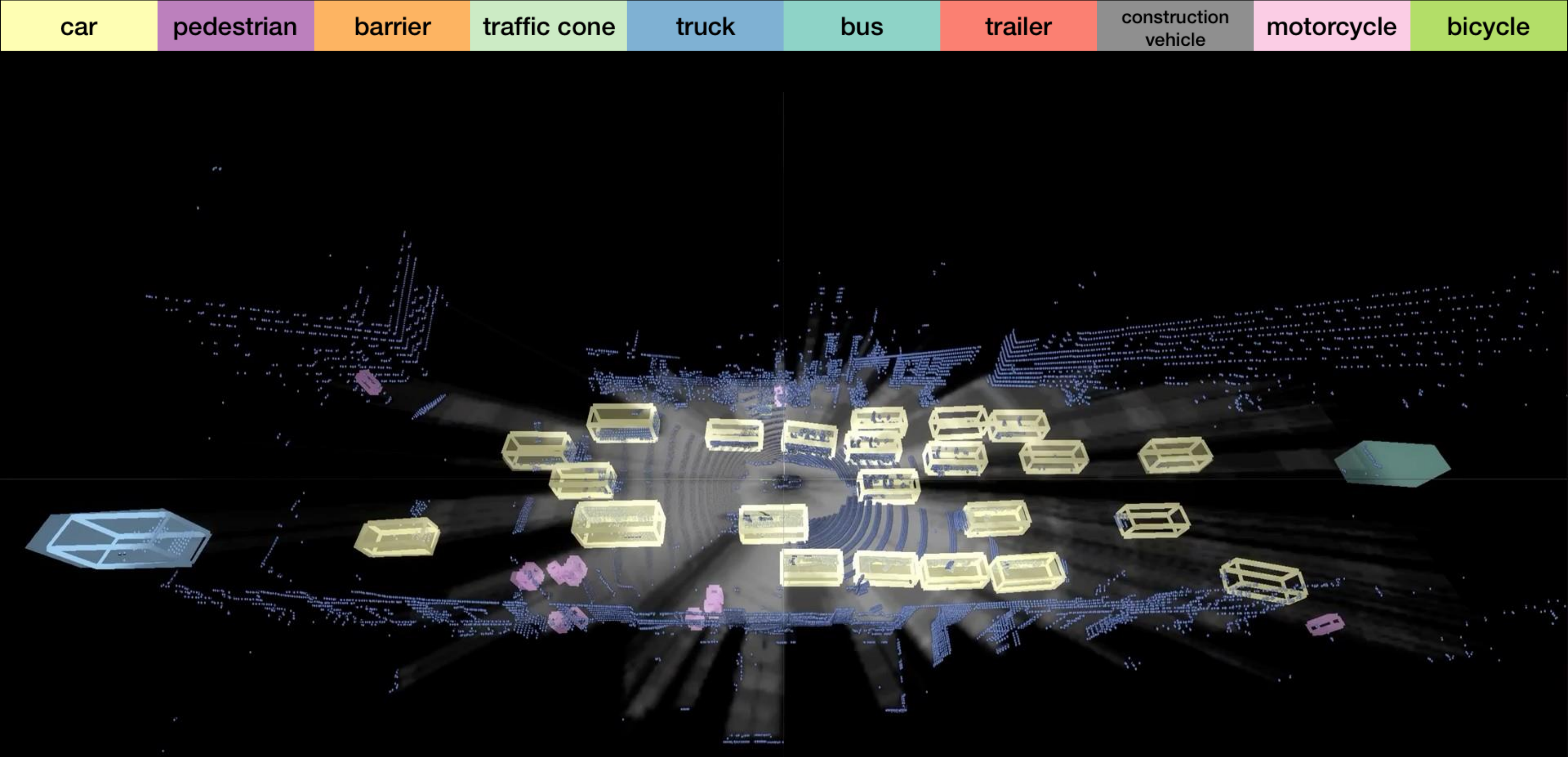
SECOND, Yan et al., Sensors'18



Visibility-aware Object Augmentation

Improve PointPillars by 4.5% in overall mAP





https://cs.cmu.edu/~peiyunh/wysiwyg

Outline

- What is lidar?
- How do we make decisions about point clouds?
  - PointNet – orderless point processing
  - VoxelNet – voxel-based point processing
  - PointPillars – bird’s eye view point processing
    - Exploiting Visibility for 3D Object Detection
  - LaserNet – range image point processing
- PseudoLidar – Bird’s eye view depth map processing

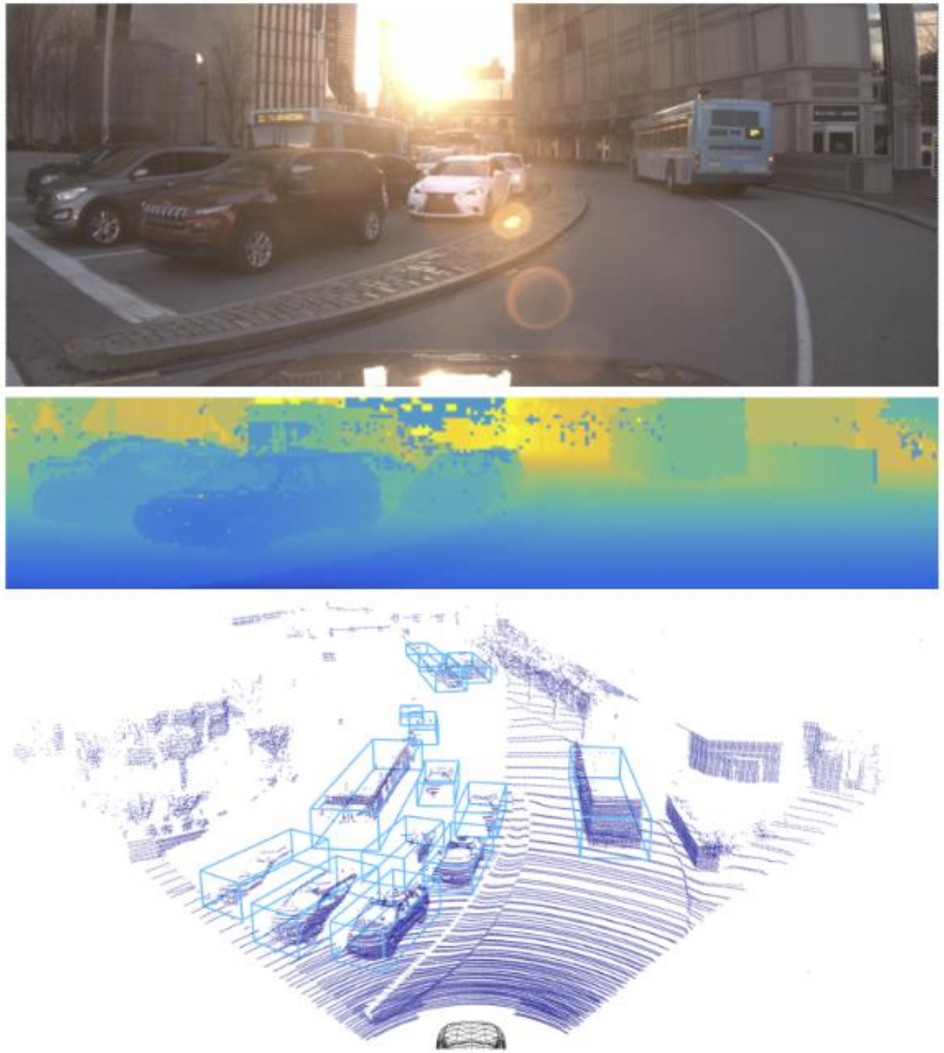


Table 4: BEV Object Detection Performance on KITTI

Method	Input	Vehicle $AP_{0.7}$ Easy	Moderate	Hard
LaserNet (Ours)	LiDAR	78.25	73.77	66.47
PIXOR [28]	LiDAR	81.70	77.05	72.95
PIXOR++ [27]	LiDAR	**89.38**	83.70	**77.97**
VoxelNet [30]	LiDAR	89.35	79.26	77.39
MV3D [5]	LiDAR+RGB	86.02	76.90	68.49
AVOD [15]	LiDAR+RGB	88.53	83.79	77.90
F-PointNet [22]	LiDAR+RGB	88.70	84.00	75.33
ContFuse [17]	LiDAR+RGB	88.81	**85.83**	77.33

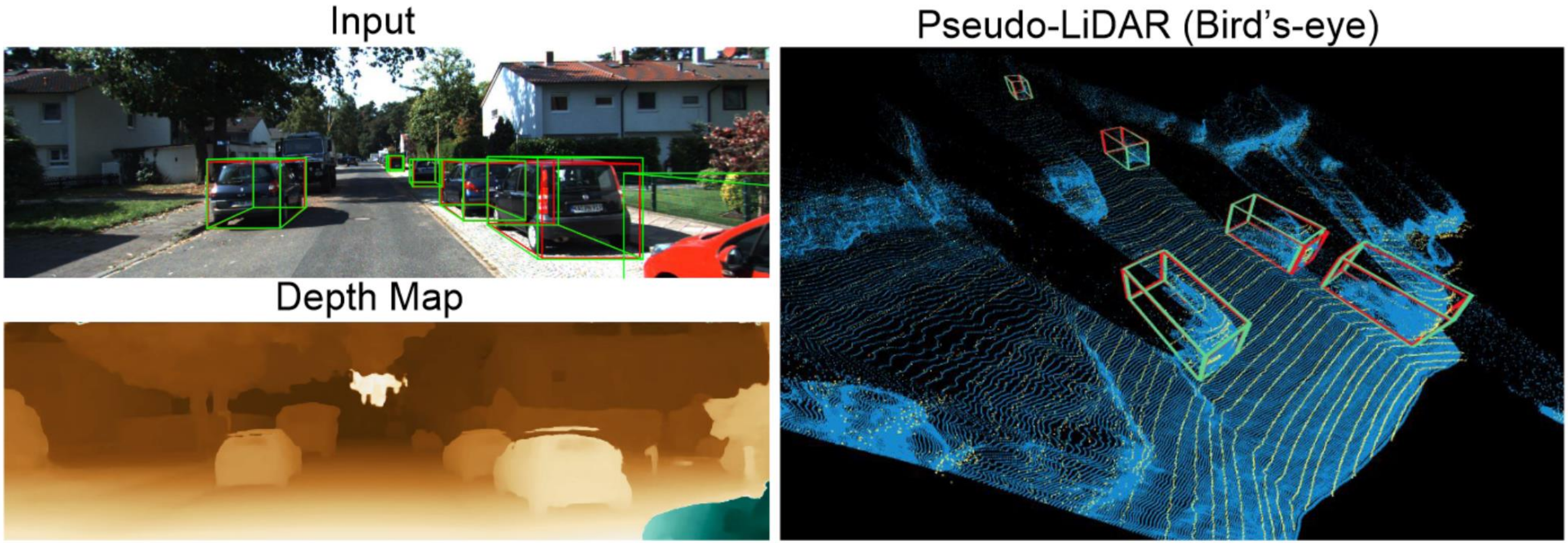
Table 3: Runtime Performance on KITTI

Method	Forward Pass (ms)	Total (ms)
LaserNet (Ours)	**12**	**30**
PIXOR [28]	35	62
PIXOR++ [27]	35	62
VoxelNet [30]	190	225
MV3D [30]	-	360
AVOD [15]	80	100
F-PointNet [22]	-	170
ContFuse [17]	60	-

LaserNet: An Efficient Probabilistic 3D Object Detector for Autonomous Driving
Gregory P. Meyer*, Ankit Laddha*, Eric Kee, Carlos Vallespi-Gonzalez, Carl K. Wellington
Uber Advanced Technologies Group. CVPR 2019

Outline

- What is lidar?
- How do we make decisions about point clouds?
    - PointNet – orderless point processing
    - VoxelNet – voxel-based point processing
    - PointPillars – bird's eye view point processing
        - Exploiting Visibility for 3D Object Detection
    - LaserNet – range image point processing
- PseudoLidar – Bird's eye view depth map processing



Pseudo-LiDAR from Visual Depth Estimation: Bridging the Gap in 3D Object Detection for Autonomous Driving
Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q. Weinberger.
CVPR 2019

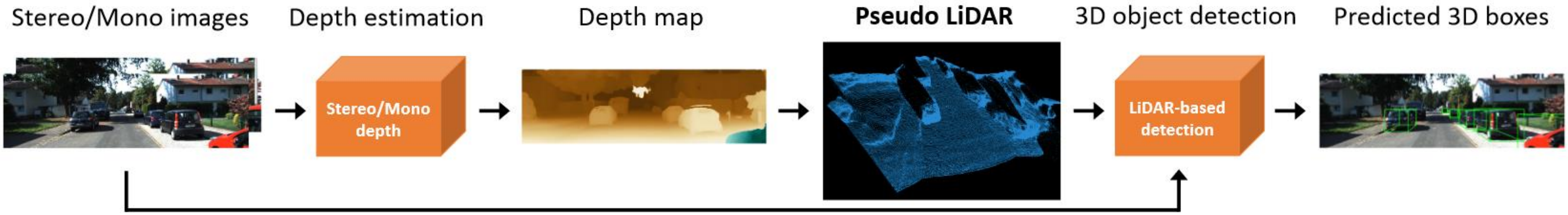
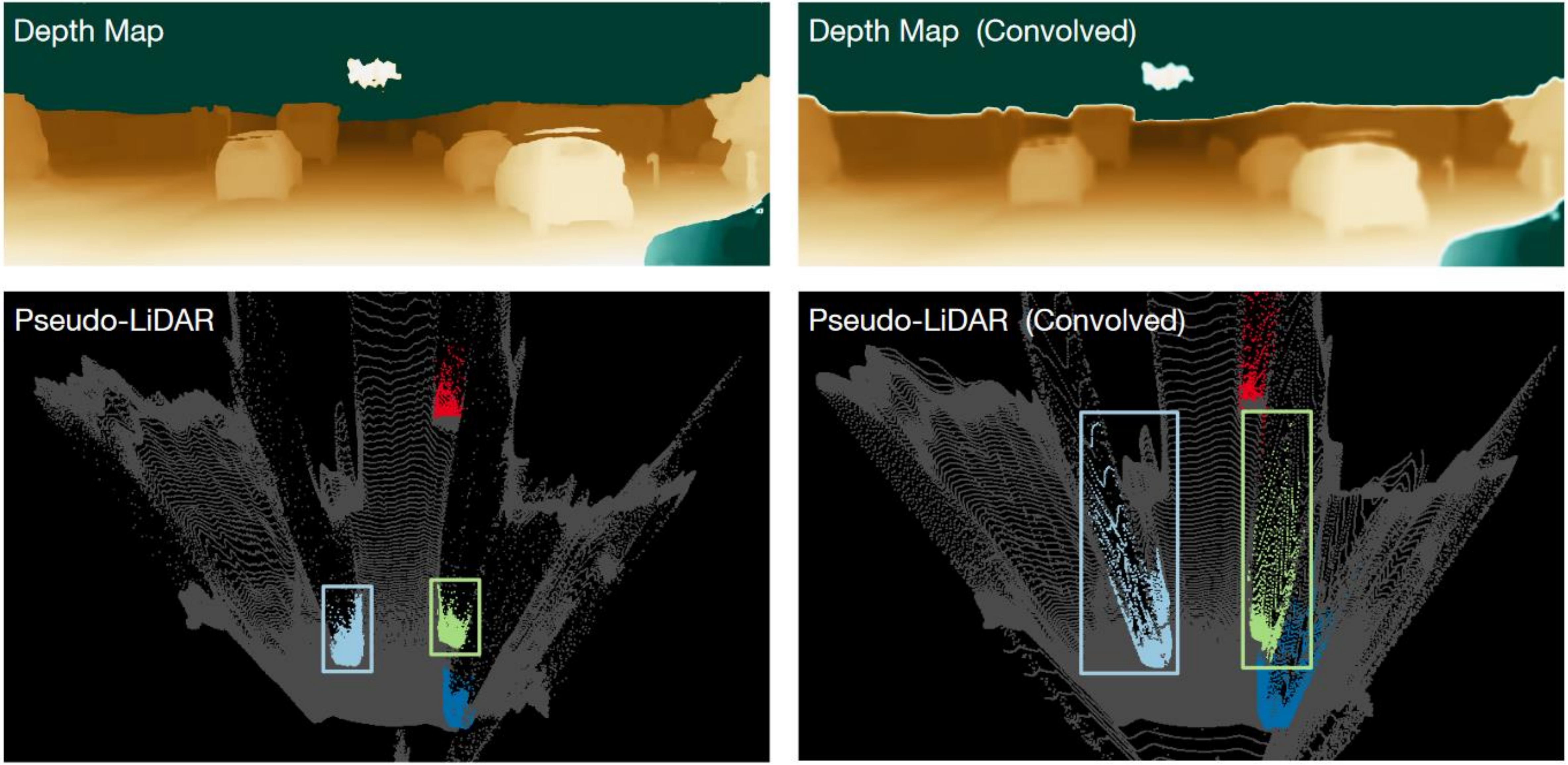


Table 1: 3D object detection results on the KITTI validation set. We report AP_{BEV} / AP_{3D} (in %) of the **car** category, corresponding to average precision of the bird's-eye view and 3D object box detection. Mono stands for monocular. Our methods with *pseudo-LiDAR* estimated by PSMNET⋆ [3] (stereo) or DORN [10] (monocular) are in blue. Methods with LiDAR are in gray. Best viewed in color.

Detection algorithm	Input signal	IoU = 0.5			IoU = 0.7		
		Easy	Moderate	Hard	Easy	Moderate	Hard
MONO3D [4]	Mono	30.5 / 25.2	22.4 / 18.2	19.2 / 15.5	5.2 / 2.5	5.2 / 2.3	4.1 / 2.3
MLF-MONO [33]	Mono	55.0 / 47.9	36.7 / 29.5	31.3 / 26.4	22.0 / 10.5	13.6 / 5.7	11.6 / 5.4
AVOD	Mono	61.2 / 57.0	45.4 / 42.8	38.3 / 36.3	33.7 / 19.5	24.6 / 17.2	20.1 / 16.2
F-POINTNET	Mono	70.8 / 66.3	49.4 / 42.3	42.7 / 38.5	40.6 / 28.2	26.3 / 18.5	22.9 / 16.4
3DOP [5]	Stereo	55.0 / 46.0	41.3 / 34.6	34.6 / 30.1	12.6 / 6.6	9.5 / 5.1	7.6 / 4.1
MLF-STEREO [33]	Stereo	-	53.7 / 47.4	-	-	19.5 / 9.8	-
AVOD	Stereo	89.0 / 88.5	77.5 / 76.4	68.7 / 61.2	74.9 / 61.9	56.8 / 45.3	49.0 / 39.0
F-POINTNET	Stereo	89.8 / 89.5	77.6 / 75.5	68.2 / 66.3	72.8 / 59.4	51.8 / 39.8	44.0 / 33.5
AVOD [17]	LiDAR + Mono	90.5 / 90.5	89.4 / 89.2	88.5 / 88.2	89.4 / 82.8	86.5 / 73.5	79.3 / 67.1
F-POINTNET [25]	LiDAR + Mono	96.2 / 96.1	89.7 / 89.3	86.8 / 86.2	88.1 / 82.6	82.2 / 68.8	74.0 / 62.0



Depth Map
Depth Map (Convolved)
Pseudo-LiDAR
Pseudo-LiDAR (Convolved)

Summary

- Popular CNN backbones aren’t a direct fit for 3D point processing tasks.
- It’s not clear how to use deep learning on 3D data
  - Use a truly permutation invariant representation (PointNet)
  - Use a voxel representation (VoxelNet)
  - Use a bird’s a view representation (PointPillars)
  - Create a range image (LaserNet)
- These alternate representations might be applicable more broadly, e.g. reasoning about depth estimates might be easier in bird’s eye view (PseudoLidar)