



**EGG**  
AMSTERDAM



data  
iku

AMSTERDAM | NOVEMBER 27

The Human-Centred  
AI Conference



POWERED BY



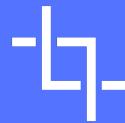
data  
iku

## 3D Object Detection from LiDAR data

Shubhabrata Roy  
HERE Technologies



# What is Point Cloud ?



Collection of points that represent a 3D shape or feature



Each point has its own set of X, Y and Z coordinates



In some cases additional attributes like reflectance etc.



Most often created by methods used in photogrammetry or remote sensing

LiDAR sensors are very common tool to collect information about the shape and feature of objects

# Attributes



Unlike 2D pixel arrays (images) or 3D voxel arrays, point clouds have an unstructured representation in that the data is simply a collection (more specifically, a set) of the points captured during a lidar or radar sensor scan.

# LiDAR



“Light Detection and Ranging”



Mounted to aerial vehicles to send laser pulses to the Earth's surface and once the laser returns back to the sensor, it will record data based on information received

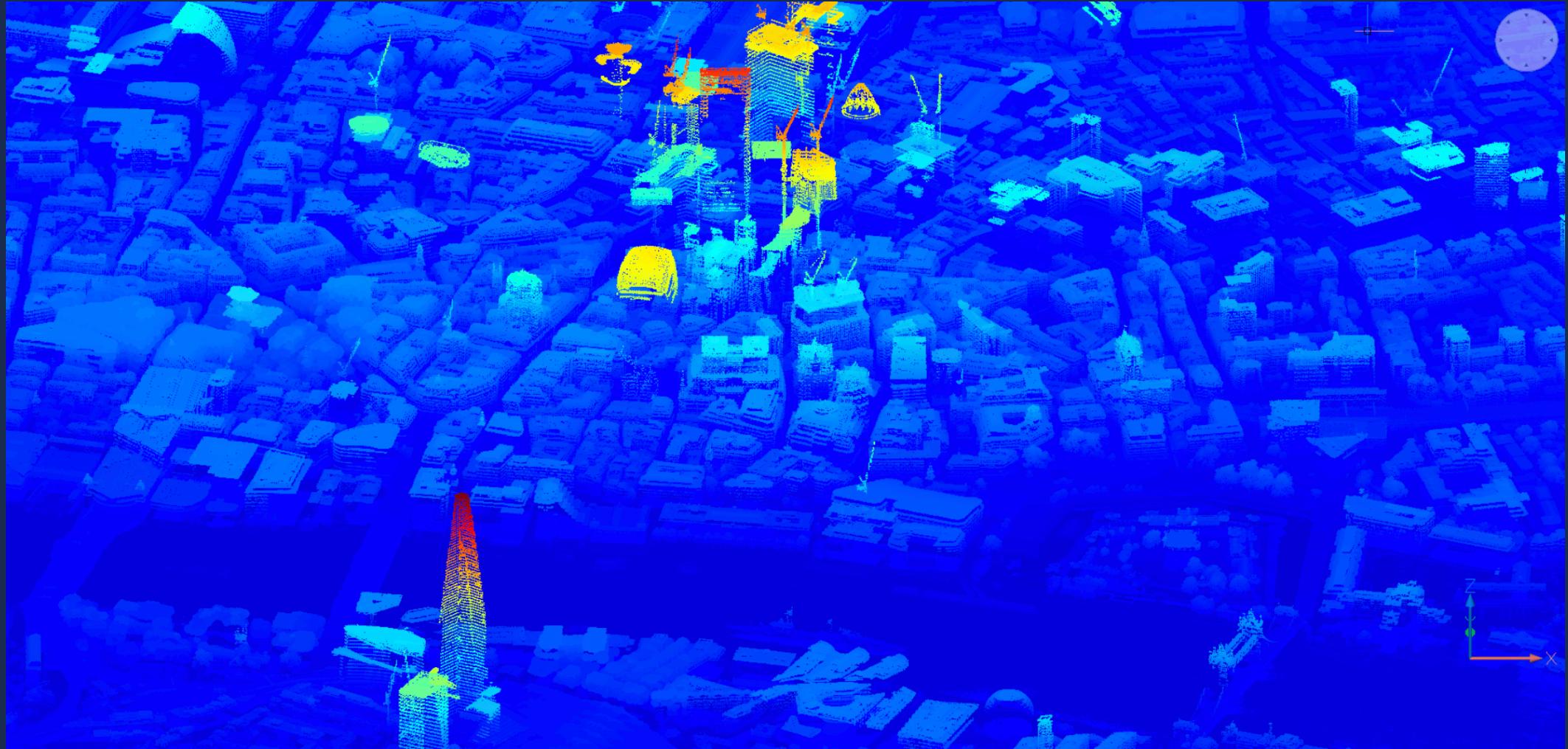


Requires many components to record data, like a GPS, so that each point will have accurate data attached

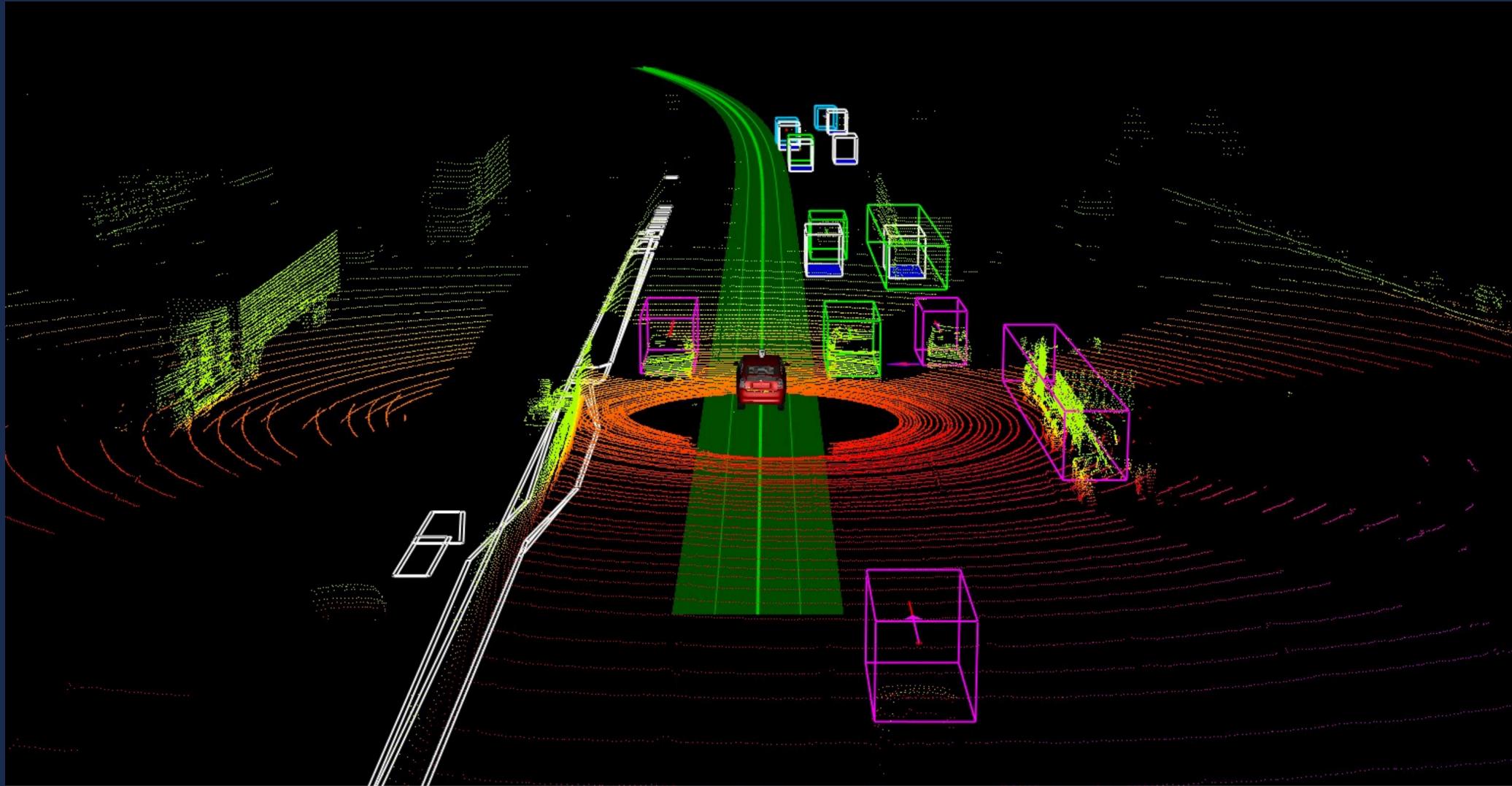


Intensity, flight angle, or even class may be recorded

# A sample point Cloud



# A sample point Cloud



# Why is it useful ?



Save surveying time



Accurate Results



Detailed object (ex. building) analysis



Design assessment for renovation

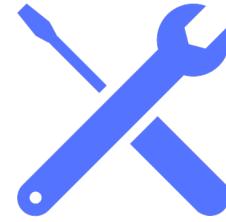


Seamless collaboration between multiple teams



Reducing overall cost

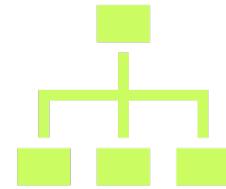
# Traditional ways to extract features from Point Cloud



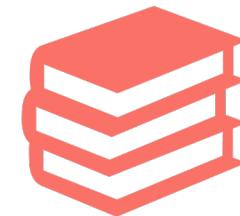
Polygonal mesh



Point-sampled  
geometry



Clustering



pcl library

# Issues with traditional approach

Geometric features, extracted from point clouds can be used with traditional ML for semantic segmentation, but it would lag from the aspect of:

- Robustness – might need heuristics-based approach for finding the bounding box
- Scalability – the proposed heuristics might not scale up
- Not Generic – require separate geometric model for different types of objects

From 3D point cloud to polygonal mesh surface, followed by some post processing

# Deep Learning based approaches



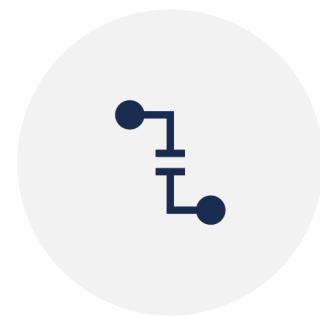
GENERALIZED



DOES NOT NEED SEPARATE  
ALGORITHMS FOR INDIVIDUAL  
FEATURES



MIGHT NOT ALWAYS REQUIRE  
2D TRANSFORMATION OR  
VOXELIZATION



SCALABLE IF APPLIED ON  
PROPERLY TRANSFORMED 3D  
POINT CLOUD (2D BEV OR RV)

# More on DL based approaches



Project the 3D cloud into 2D top-down Bird Eye View



Project the 3D cloud into 2D top-down Range View

Apply 2D convolution with other DL methods or heuristics-based approaches



Discretizing the LiDAR point cloud into 3D voxel



Directly use 3D point cloud data

Apply 3D convolution

## Bird Eye View vs Rear View

- ✓ Preserves object size with range
- ✓ Provides a strong prior for learning
- ✓ The Z-axis is treated as a feature channel for 2D convolutions
- ✓ Suffers from occlusion and object size variation w.r.t. distance
- ✓ Low latency due to the dense representation compared to the sparser bird eye view

# Some relevant NN Architectures

2D

- R-CNN
- Fast R-CNN
- Faster R-CNN
- Mask R-CNN
- Cascade R-CNN

3D

- PointNet
- PointNet++
- Point R-CNN

# Some Practical Issues of object detection



Draw a bounding box around the object of interest to locate it within the image



There could be many bounding boxes representing different objects of interest within the image (not know how many beforehand)



Take different regions of interest from the image and use a CNN to classify the presence of the object within that region

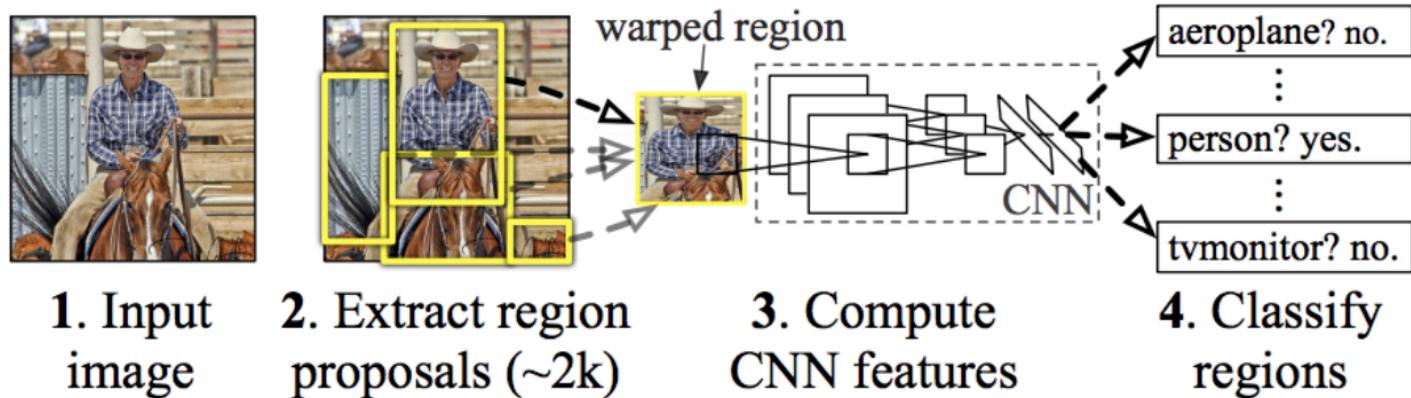


But objects of interest might have different spatial locations within the image and different aspect ratios



Need to select a huge number of regions and this could computationally blow up

## R-CNN: *Regions with CNN features*



R-CNN

- Use selective search to extract just 2000 regions from the image -region proposals
- Selective Search:
  - Generate initial sub-segmentation, we generate many candidate regions
  - Use greedy algorithm to recursively combine similar regions into larger ones
  - Use the generated regions to produce the final candidate region proposals

# R-CNN - issues



Still takes a huge amount of time to train the network as you would have to classify 2000 region proposals per image

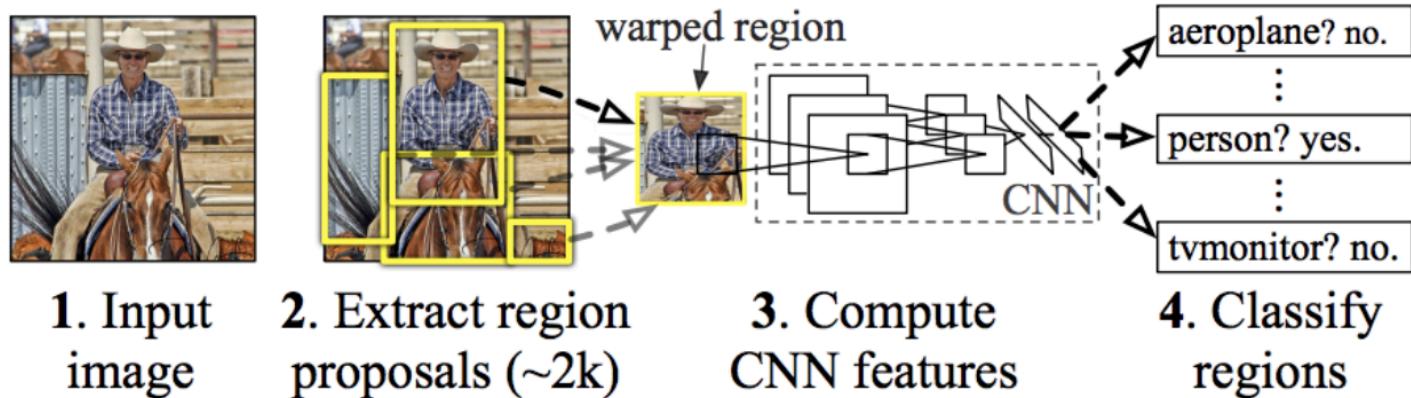


Can not be implemented real time as it takes around 47 seconds for each test image.



The selective search algorithm is a fixed algorithm. Therefore, no learning is happening at that stage. This could lead to the generation of bad candidate region proposals.

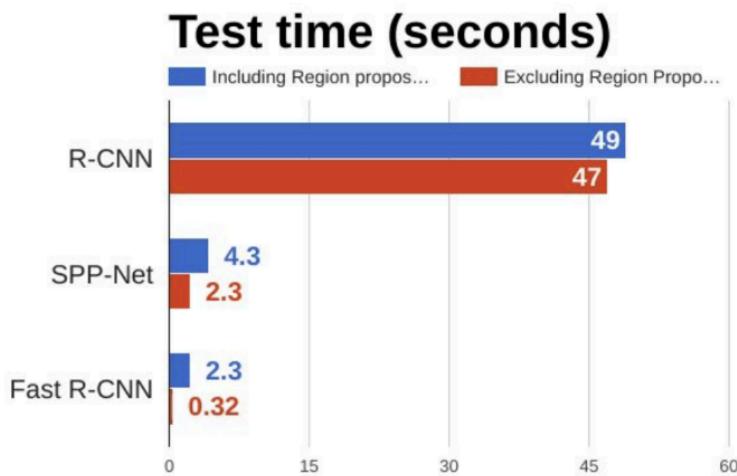
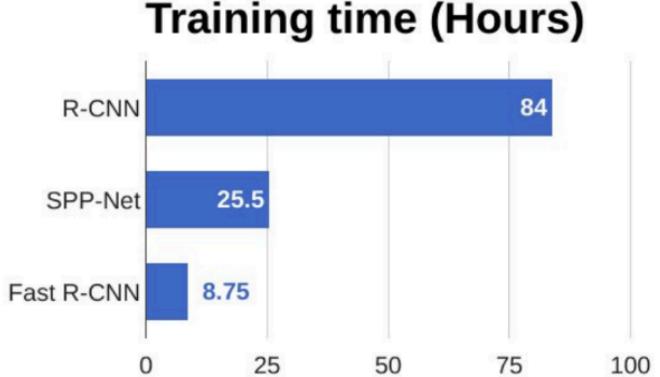
## R-CNN: *Regions with CNN features*



R-CNN

## Fast R-CNN

- Feed the input image to the CNN to generate a convolutional feature map
- Identify the region of proposals
- Warp them into squares and by using a RoI pooling layer reshape them into a fixed size so that it can be fed into a fully connected layer
- From the RoI feature vector, use a softmax layer to predict the class of the proposed region (also the offset values for the bounding box)



Comparison of object detection algorithms

## Fast R-CNN

- The reason “Fast R-CNN” is faster than R-CNN - you don’t have to feed 2000 region proposals to the convolutional neural network every time
- The convolution operation is done only once per image and a feature map is generated from it.
- However, including region proposals slows down the algorithm significantly when compared to not using region proposals

# Faster R-CNN

---

Selective search is a slow and time-consuming process affecting the performance of the network

---

Faster R-CNN eliminates the selective search algorithm and lets the network learn the region proposals

---

Image is provided as an input to a convolutional network which provides a convolutional feature map

---

Instead of using selective search algorithm on the feature map to identify the region proposals, a separate network is used to predict the region proposals

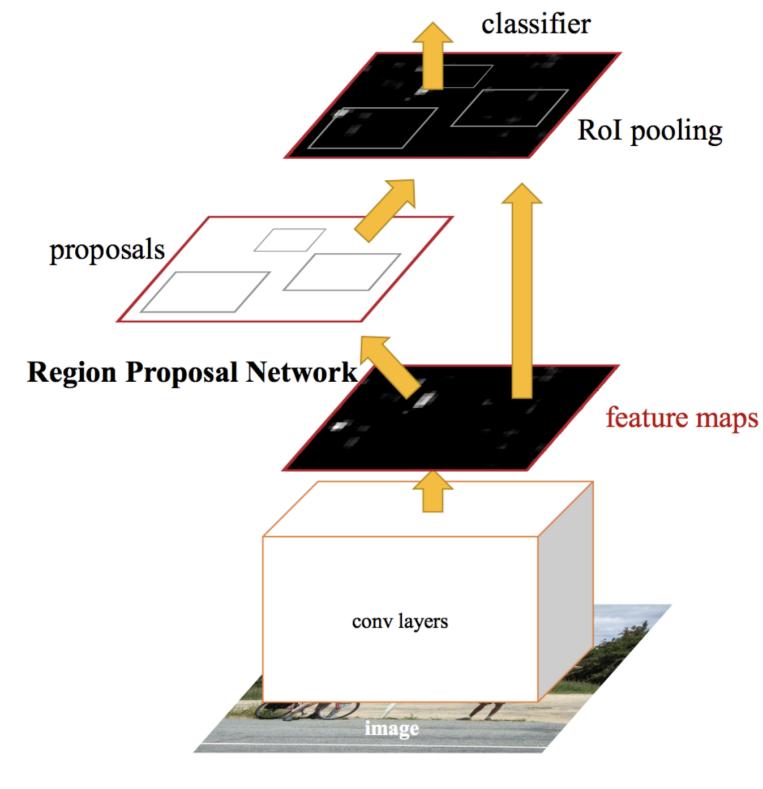
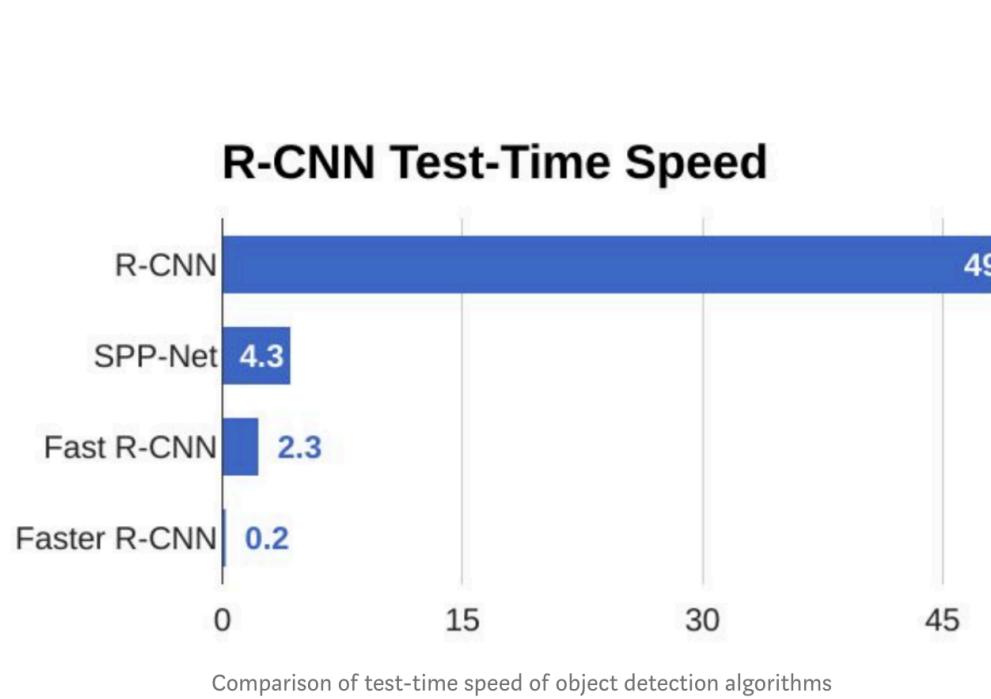
---

The predicted region proposals are reshaped using a RoI pooling layer

---

Classify the image within the proposed region and predict the offset values for the bounding boxes.

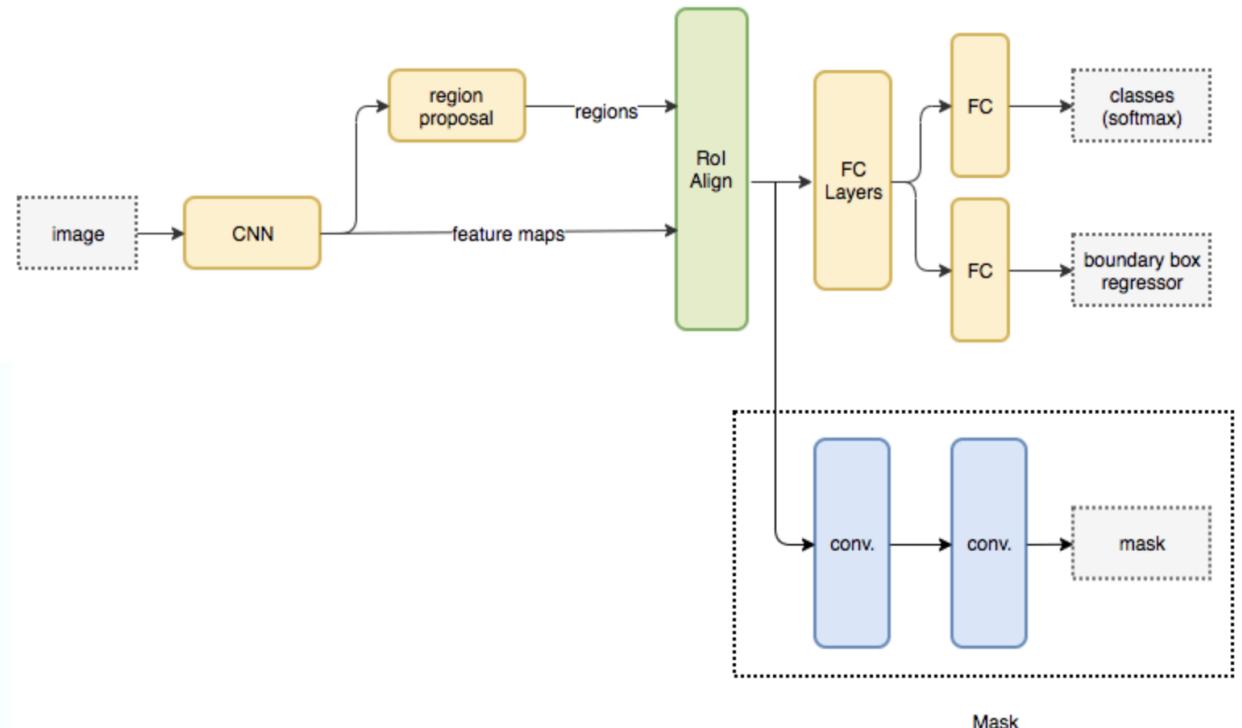
# Faster R-CNN



# Mask R-CNN

- The Faster R-CNN builds all the ground works for feature extractions and ROI proposals for Mask R-CNN
- After the ROI pooling, 2 more convolution layers are added to build the mask

ROI proposals. At first sight, performing image segmentation may require more detail analysis to colorize the image segments. By surprise, not only



# Mask R-CNN

Another major contribution of Mask R-CNN is the refinement of the ROI pooling

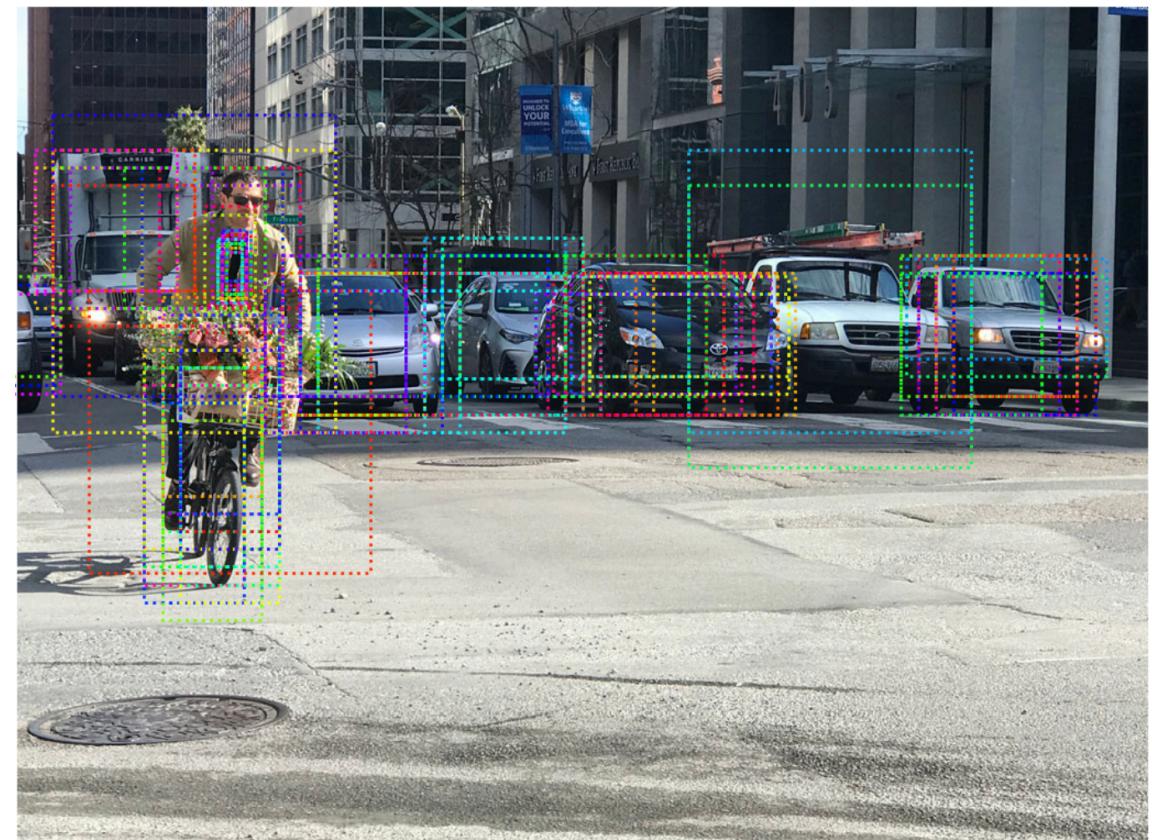
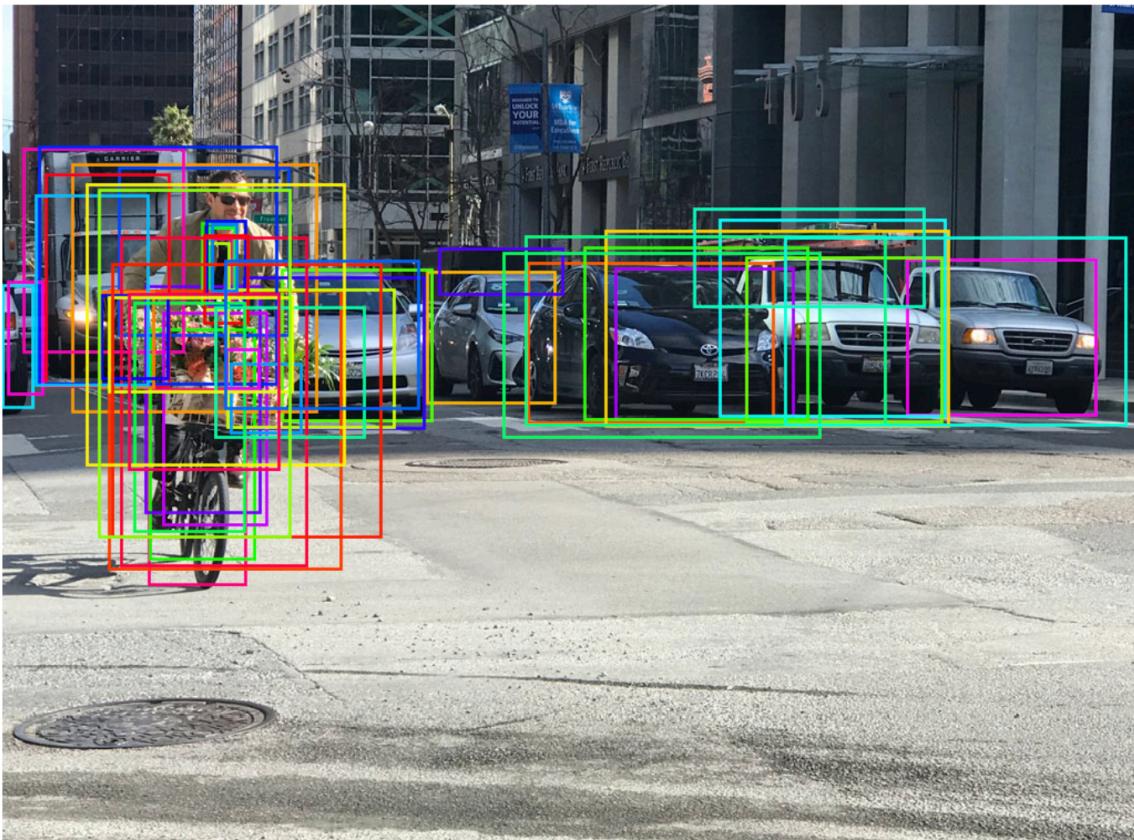
0.1	0.3	0.2	0.3	0.2	0.6	0.8	0.9
0.4	0.5	0.1	0.4	0.7	0.1	0.4	0.3
0.2	0.1	0.3	0.8	0.6	0.2	0.1	0.1
0.4	0.6	0.2	0.1	0.3	0.6	0.1	0.2
0.1	0.8	0.3	0.3	0.5	0.3	0.3	0.3
0.2	0.9	0.4	0.5	0.1	0.1	0.1	0.2
0.3	0.1	0.8	0.6	0.3	0.3	0.6	0.5
0.5	0.5	0.2	0.1	0.1	0.2	0.1	0.2

0.1	0.3	0.2	0.3	0.2	0.6	0.8	0.9
0.4	0.5	0.1	0.4	0.7	0.1	0.4	0.3
0.2	0.1	0.3	0.8	0.6	0.2	0.1	0.1
0.4	0.6	0.2	0.1	0.3	0.6	0.1	0.2
0.1	0.8	0.3	0.3	0.5	0.3	0.3	0.3
0.2	0.9	0.4	0.5	0.1	0.1	0.1	0.2
0.3	0.1	0.8	0.6	0.3	0.3	0.6	0.5
0.5	0.5	0.2	0.1	0.1	0.2	0.1	0.2

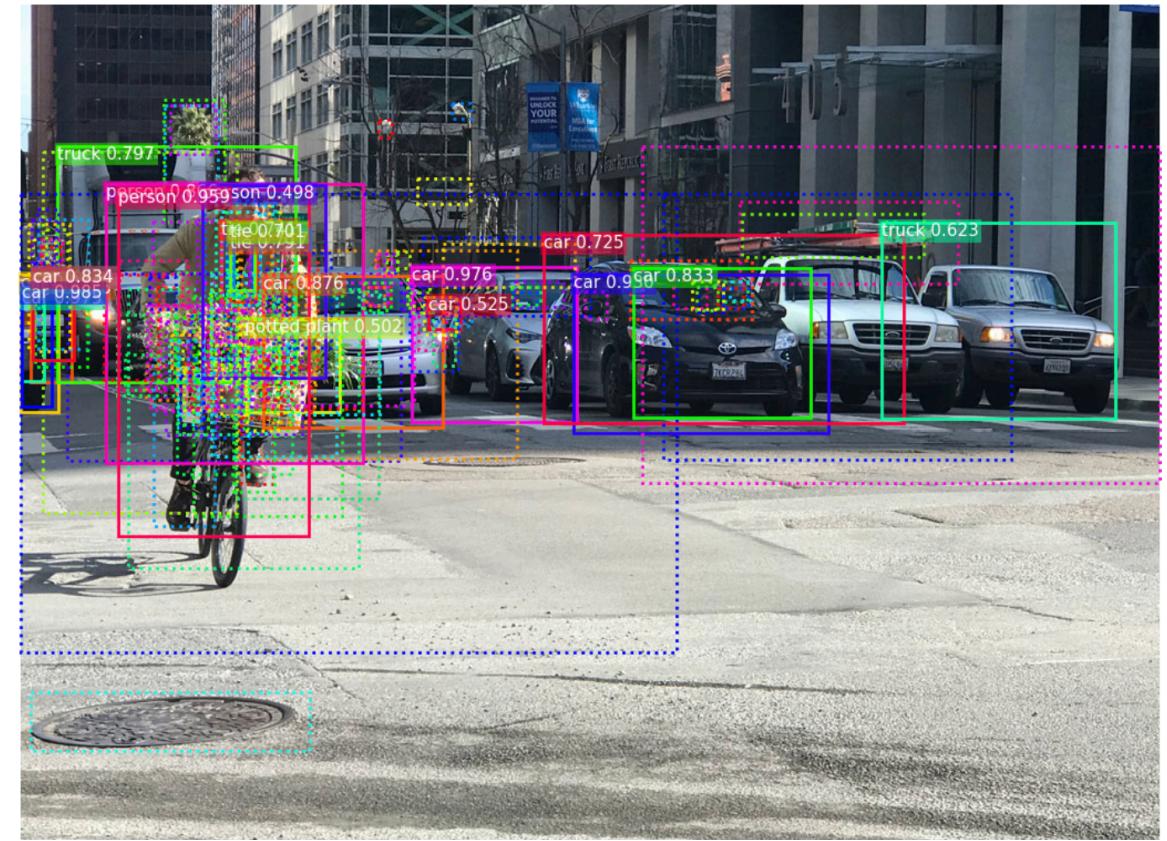
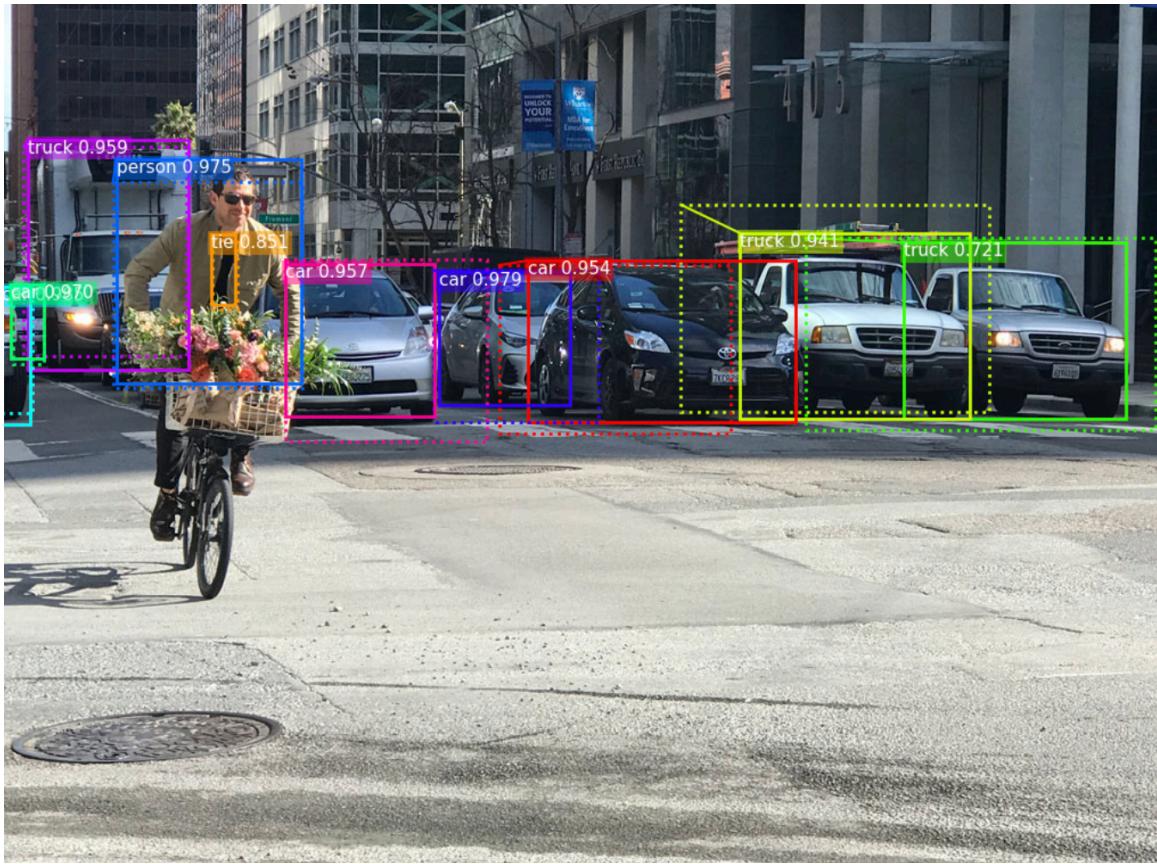
0.8	0.6
0.9	0.6

0.88	0.6
0.9	0.6

# Mask R-CNN



# Mask R-CNN



# Cascade R-CNN



In R-CNN based approaches detection is framed as a multi-task learning problem combining classification and bounding box regression



An intersection over union (IoU) threshold is required to define positives/negatives in object detection



Object detector is trained with low IoU threshold- produces noisy detections



Detection performance tends to degrade with increasing the IoU thresholds

Overfitting during training, due to exponentially vanishing positive samples, and Inference-time mismatch between the IoUs for which the detector is optimal

# Cascade R-CNN

Cascade R-CNN, is proposed to address these problems

It consists of a sequence of detectors trained with increasing IoU thresholds, to be sequentially more selective against close false positives

Detectors are trained stage by stage, leveraging the observation that the output of a detector is a good distribution for training the next higher quality detector

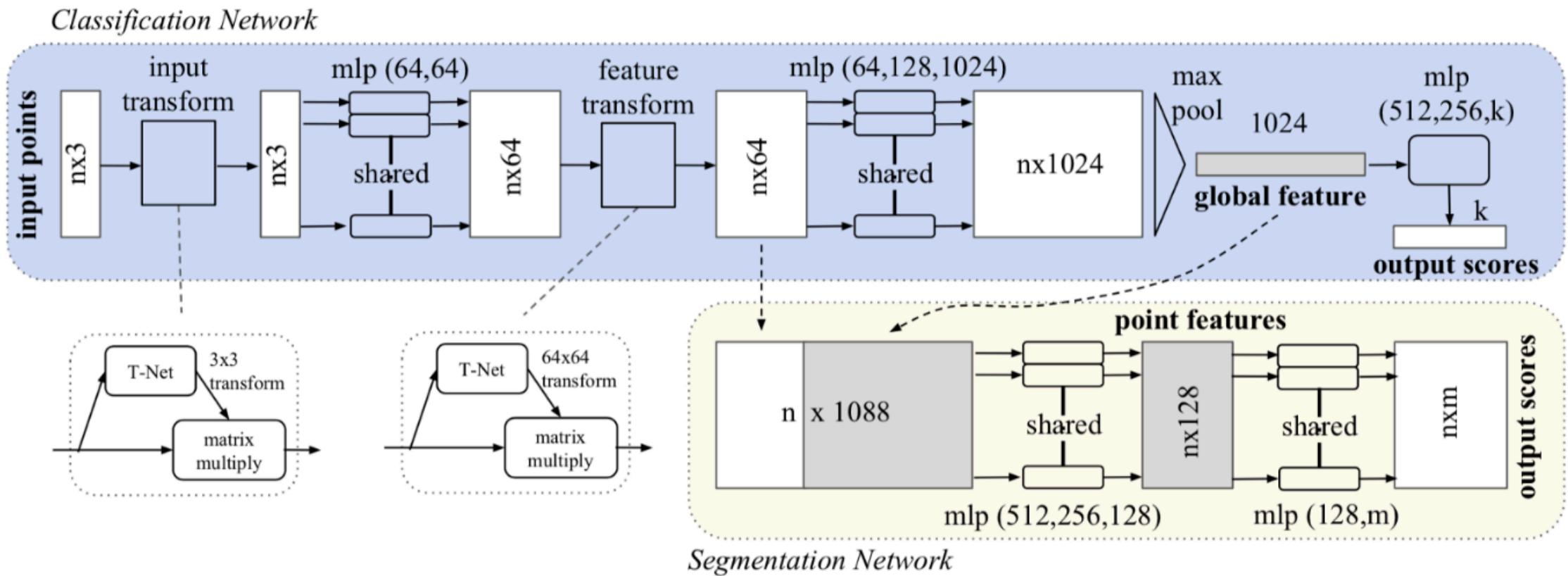
# PointNet

So far, we discretized a point cloud by taking multi-view projections onto 2D to leverage existing techniques built around 2D convolutions,

Given that the original data is manipulated, it can have negative impact

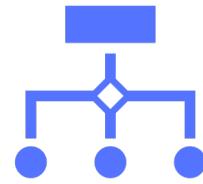
PointNet Takes raw point cloud data as input, which is typically collected from either a lidar or radar sensor

# PointNet



# PointNet

PointNet does not really tackle the problem related to the desertion of shape



Can not capture local structure induced by the metric space points



The max pooling operation takes the full cloud as input to produce a global feature

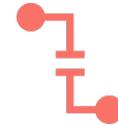


Makes it unlikely to learn fine grained patterns or to understand complex scenes

# PointNet++



Pointnet++ is built upon  
Pointnet



A hierarchical network that  
applies Pointnet recursively on  
a nested portioning of the  
input point cloud



Proposes novel set learning  
layers to adaptively combine  
features from multiple scales  
from varying densities



Like CNNs, it extracts local  
features from a small  
neighborhood and further  
groups into larger units to  
produce higher level features



This process is recursive until  
we obtain the feature of the  
whole point set

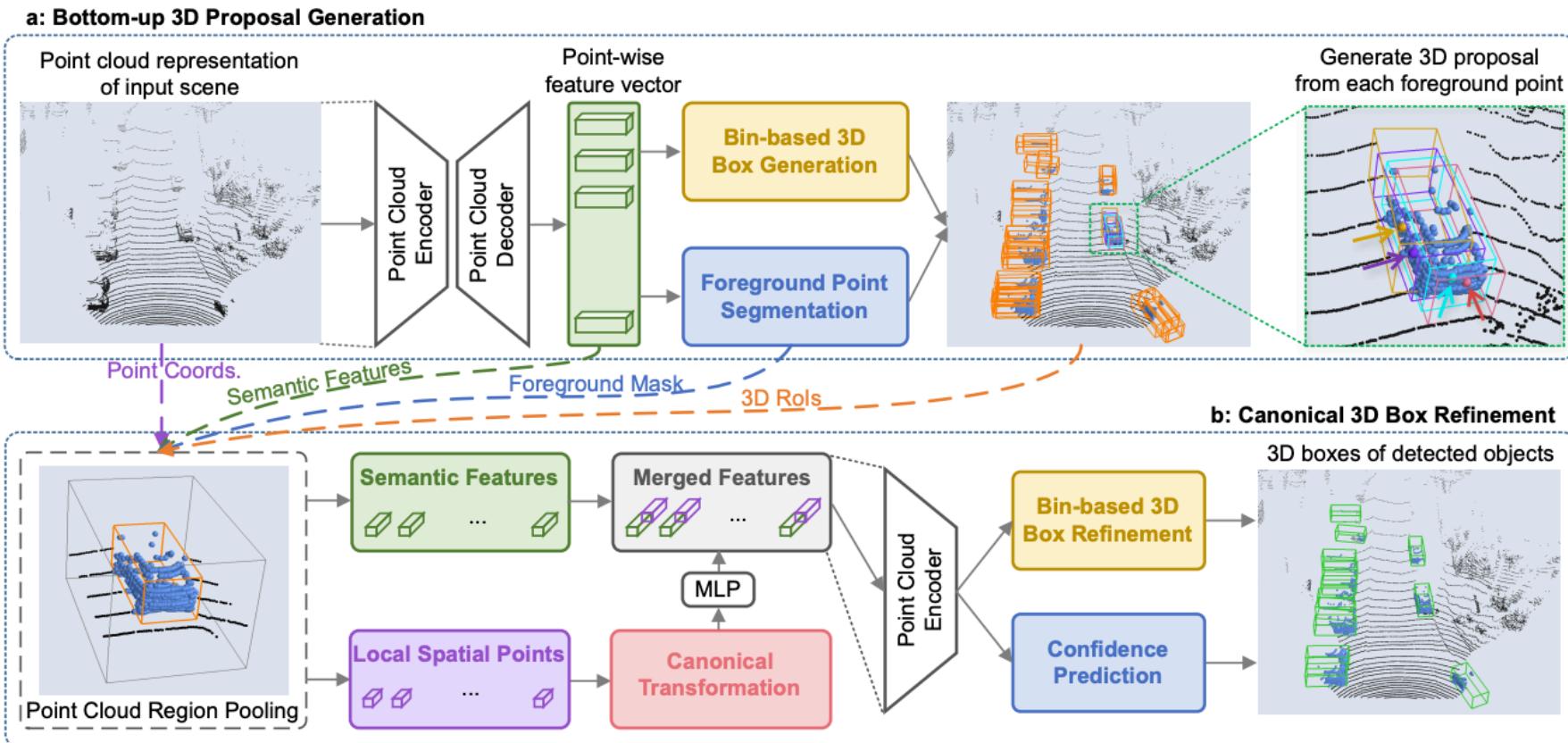
# Point R-CNN

- The whole framework is composed of two stages:
  - ✓ Stage-1: Bottom-up 3D proposal generation:

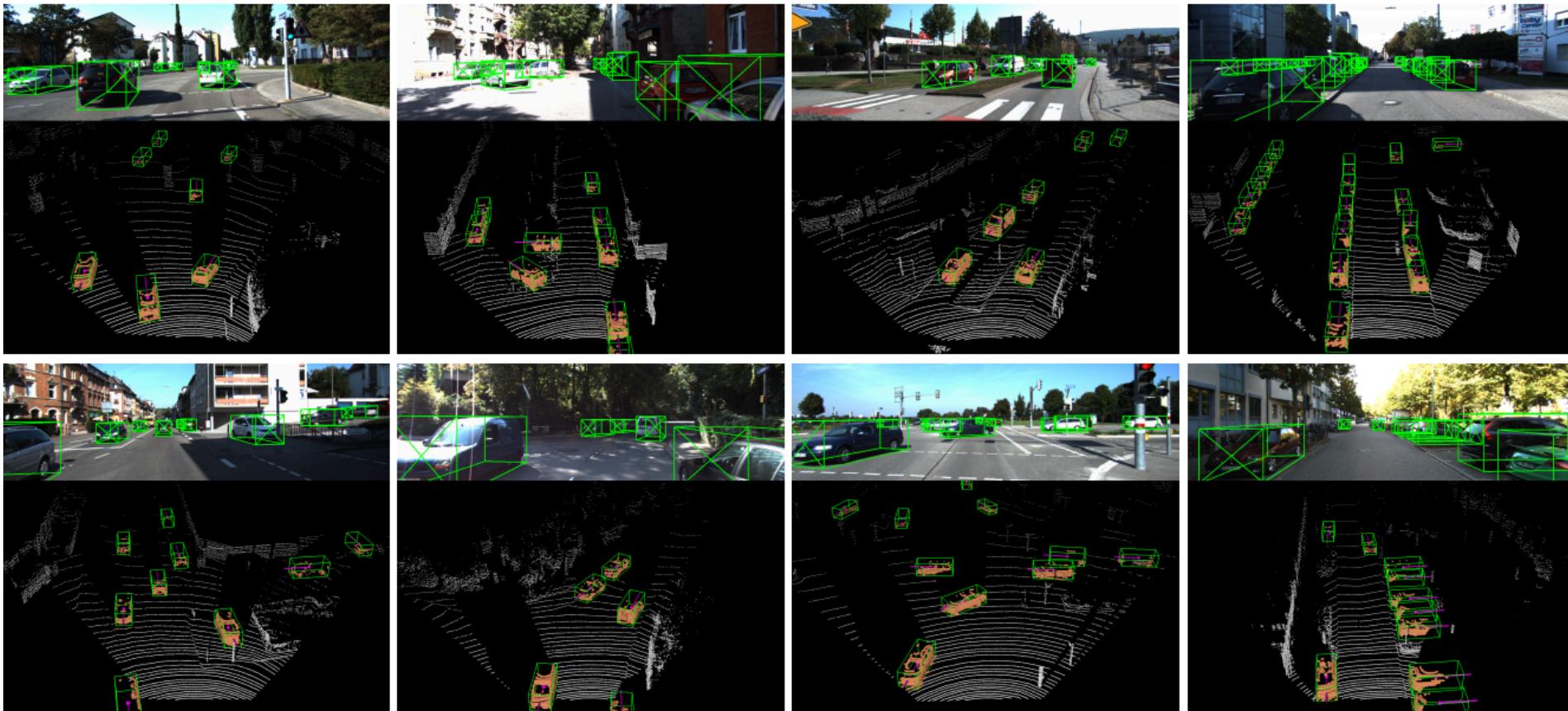
Instead of generating proposals from RGB image or projecting point cloud to bird's view or voxels, stage-1 sub-network directly generates a small number of high-quality 3D proposals from point cloud in a bottom-up manner via segmenting the point cloud of the whole scene into foreground points and background
  - ✓ Stage-2 Refining proposals to obtain the final detection results:

Transforms the pooled points of each proposal to canonical coordinates to learn better local spatial features, which is combined with global semantic features of each point learned in stage-1 for accurate box refinement and confidence prediction

# Point R-CNN



# Point R-CNN



# Summary

We introduced point cloud and some tools to analyze it

Deep learning is used widely on this type of data

2D (BEV or RV) and 3D based approach

Presented several DL based object detection approaches that work on 2D or 3D data

Discussed pros and cons of these approaches and their sequential evolution

# Conclusion



Object detection on 2D projection is simpler, however we might lose some information.



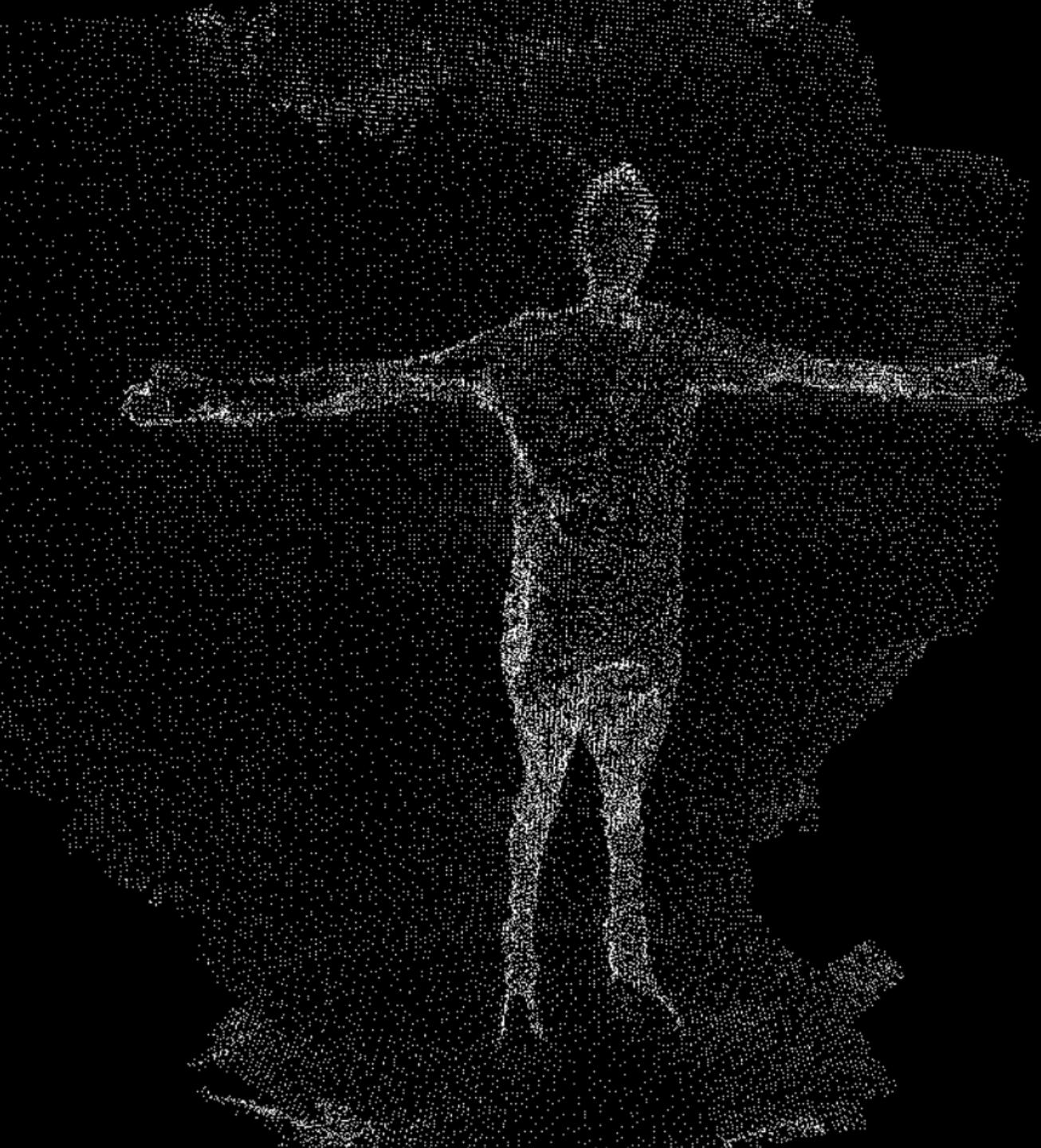
Reconstructing 3D objects after detecting 2D objects from the projections can be a bottleneck in terms of generalization and performance



Several works on 3D object detection without data transformation –can be compute intensive



Approaches like Point R-CNN can be a potential future direction



Thank You

---