

# RLLAB

Melissa Mozifian

McGill University

Real-time 3D Object Detection for Autonomous Driving

September 21, 2018



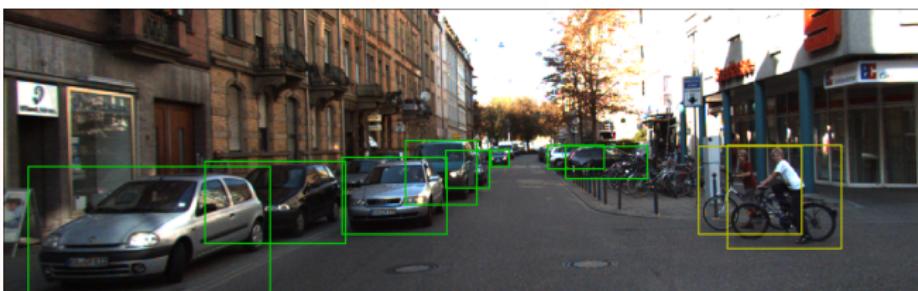


## Outline

- 1 3D Object Detection: Motivation
  - 2 Problem Background
  - 3 AVOD - Aggregate View Object Detection
  - 4 AVOD-SSD
  - 5 Discussion & Future Work

## Motivation

- The detection and localization of objects is a key problem in creating autonomous cars.
  - 2D detectors with high accuracy are deployed in consumer products.



**Figure:** Example of 2D detection on KITTI dataset [1].

## Motivation

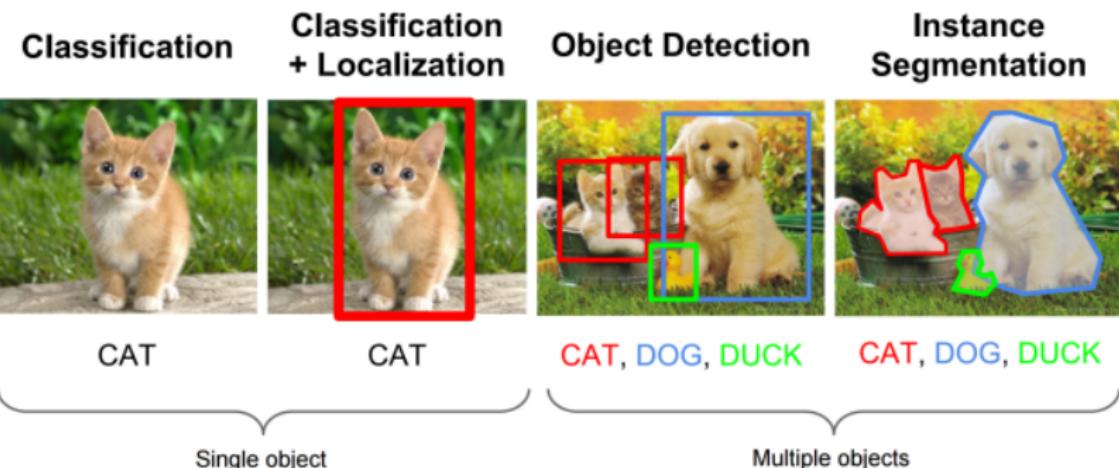
- In the context of autonomous driving, 2D bounding boxes are not sufficient:
    - Lack of 3D pose and location.
    - Occlusion.
  - The gap between 2D and 3D detections remains large due to:
    - Adding a third dimension.
    - Low resolution of 3D data.
    - Deterioration of quality of 3D data as a function of distance.

## Motivation



**Figure:** Example of 3D detection on KITTI dataset.

## Background: Object Detection



**Figure:** Example of object classification, localization and segmentation [2].

## Background: Classic Object Detectors



- Most classical methods adopted the sliding-window technique which works by sliding a box around image and classifying each image crop inside a box.
  - Pick a classifier (say SVM), run it at evenly spaced locations over the entire image, focusing on smaller areas of the feature map.

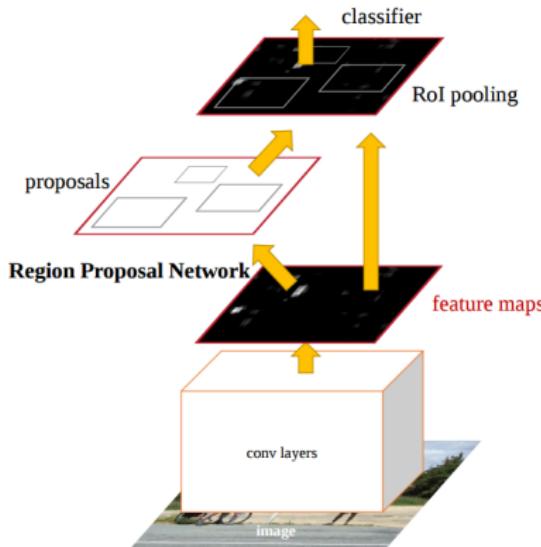
### Background: Two-Stage Detection



- The dominant paradigm in modern object detection.
  - **First stage:** Generates a set of candidate proposals that should contain all the objects.
  - **Second stage:** Classifies the object proposals and refines the location estimation of objects.

## Background: Faster R-CNN

- Region Proposal Network (RPN)
  - Region of interest pooling (RoI pooling)



**Figure:** Faster R-CNN architecture overview [3].

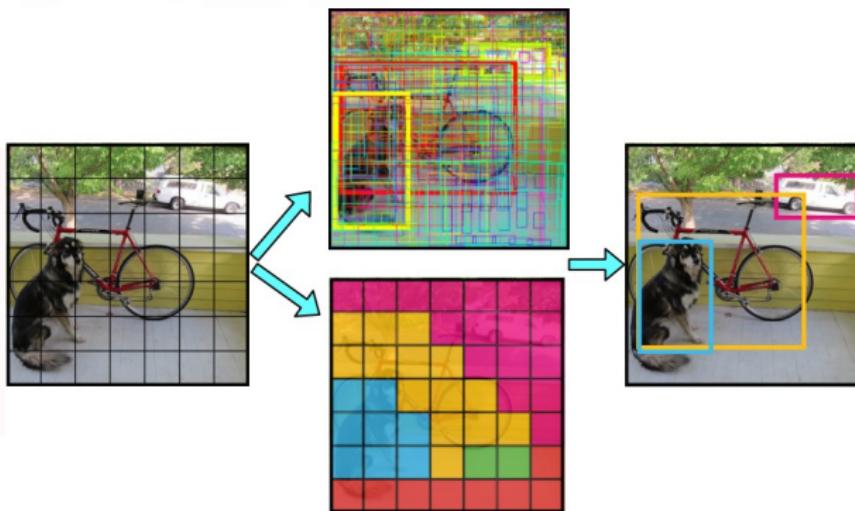


## Background: Single-Staged Detectors

- Single Stage/Shot object detectors have also been proposed as RPN free architectures.
  - Example architectures:
    - OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks (Zhang et al - 2013)
      - Multi-scale, sliding window approach
    - You Only Look Once: YOLO (Redmon et al - 2015)
      - Discretizes the image into grids and then each grid cell proposes potential bounding boxes and scores those boxes using convolutional features.
    - Single Shot Detector: SSD (Liu et al - 2016)
      - Discretizes the output space of bounding boxes into a set of default boxes over different aspect ratios and scales per feature map location

## Background: Single-Stage Detection

- RPN free architectures.
  - Example: You Only Look Once: YOLO (Redmon et al - 2015)



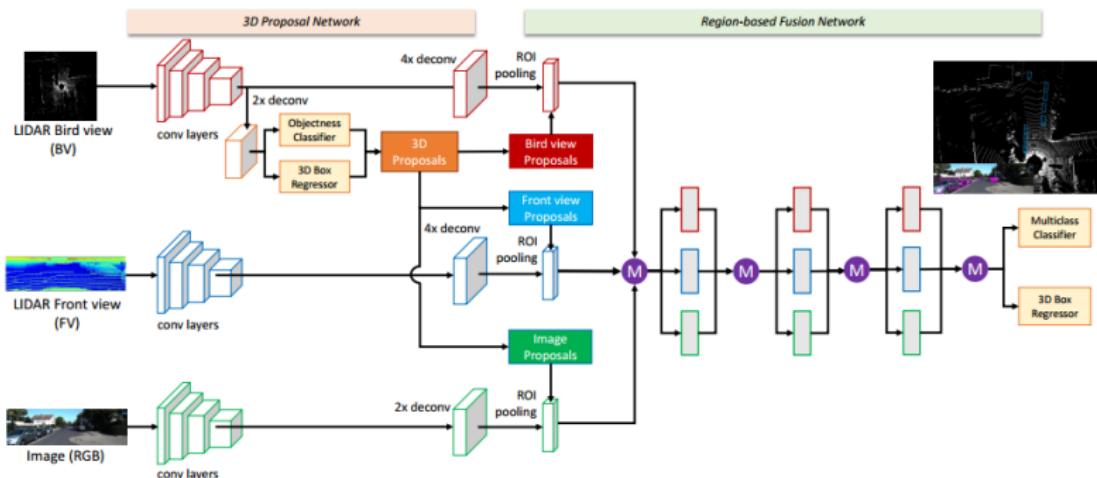
**Figure:** Yolo detection mechanism [8].

## Background: 3D Object Detection



- **3D RPN**: Generates a set of candidate 3D boxes.
  - **Second stage**: Classifies and regresses the 3D boxes.
  - Example architectures:
    - **MV3D** - Multi-View 3D Object Detection Network for Autonomous Driving. (Chen et al 2017) [5]
    - **F-PointNet** - Frustum PointNets for 3D Object Detection from RGB-D Data. (Qi et al 2018) [6]

Background: 3D Object Detector - MV3D<sup>1</sup>



<sup>1</sup> Multi-View 3D Object Detection Network for Autonomous Driving(CVPR) 2017.

## Background: MV3D

- A sensor fusion network.
  - Encodes the sparse 3D point cloud into bird's eye view.
  - An RPN generates 3D candidate boxes from the bird's eye view representation.
  - Uses 8 corner box representation to regress 3D bounding boxes.

# Background: F-PointNet

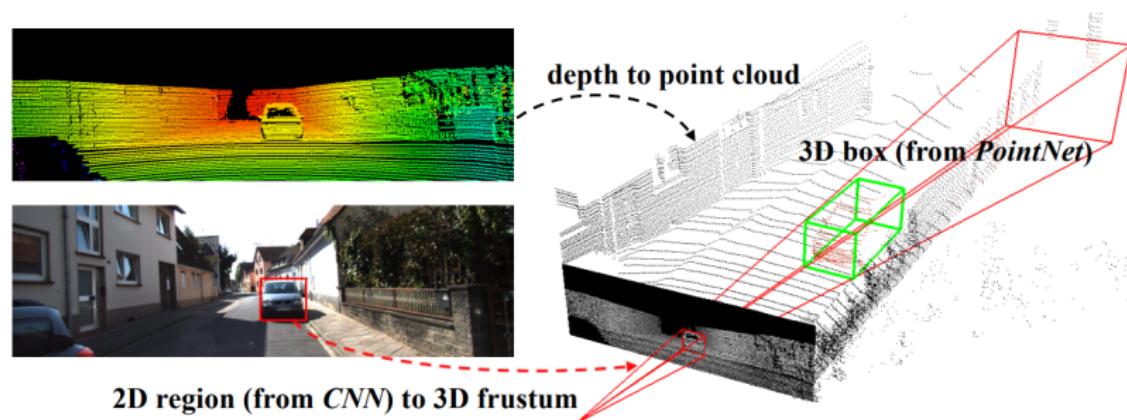
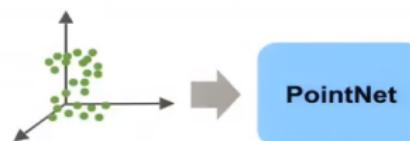


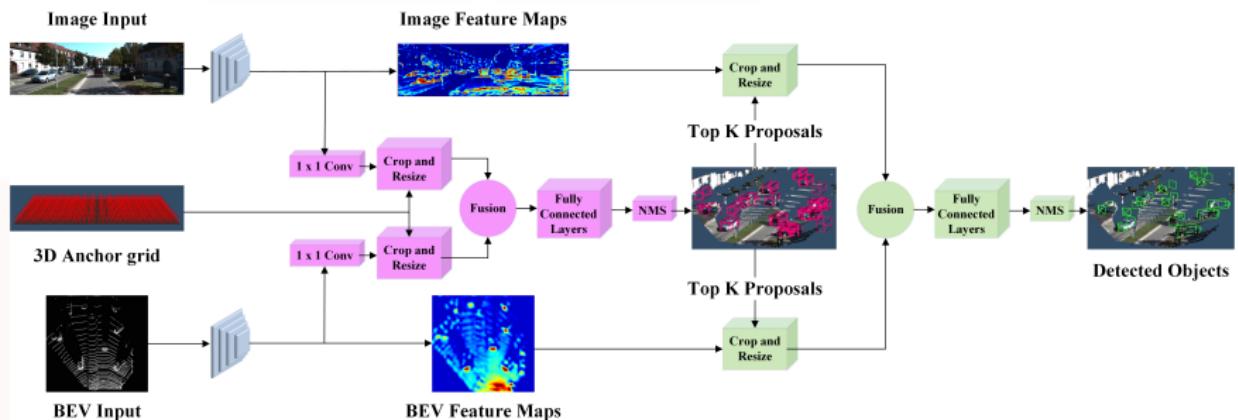
Figure: F-PointNet detection pipeline.

## Background: PointNet

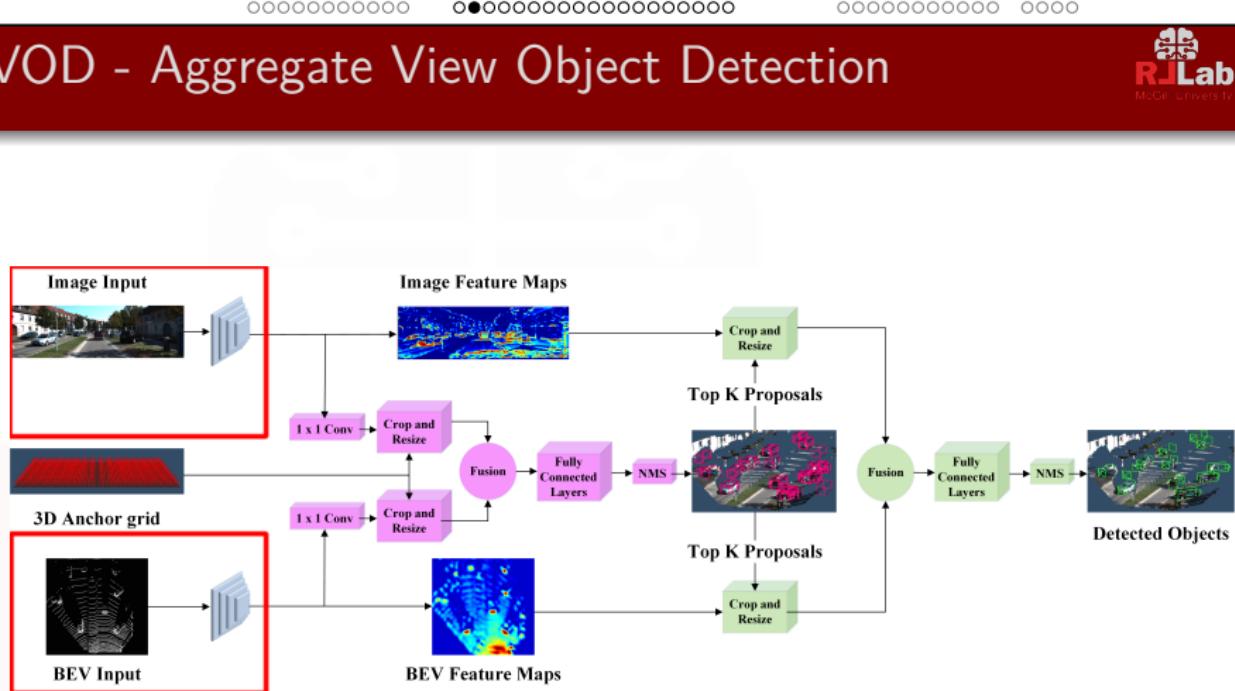
- Pointcloud feature learning
  - Avoid converting pointcloud to other representations
    - Voxelization
    - Projection/Rendering
    - Feature extraction



**Figure:** F-PointNet detection pipeline [4].



**Figure:** AVOD architectural diagram. The feature extractors are shown in **blue**, the region proposal network in **pink**, and the second stage detection network in **green**.



**Figure:** AVOD architectural diagram. The feature extractors are shown in **blue**, the region proposal network in **pink**, and the second stage detection network in **green**.

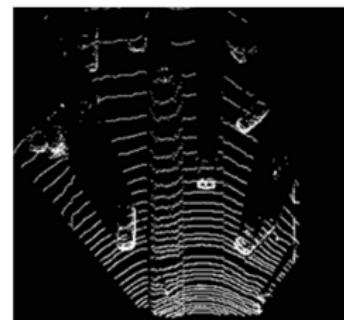
## AVOD - Input Data

- RGB Image
  - Bird's Eye View (**BEV**) Maps:
    - Small size variance.
    - Small variance in vertical location.
    - No issues with occlusion.

## Image Input



## BEV Input



# Generating Feature Maps

- VGG Feature Extractor:

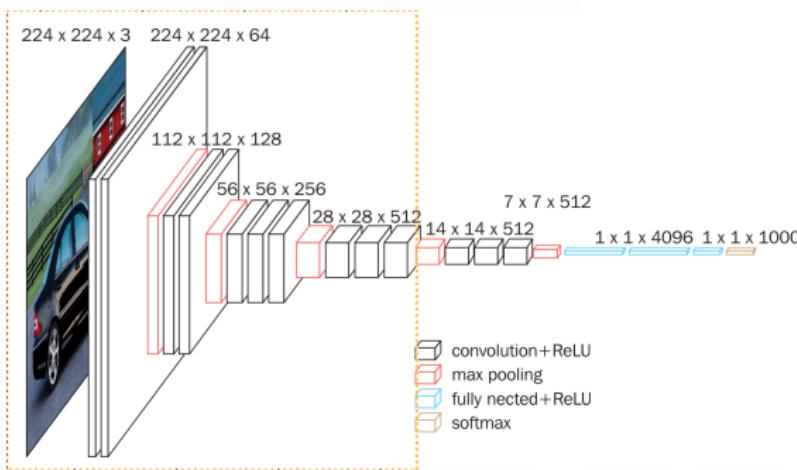
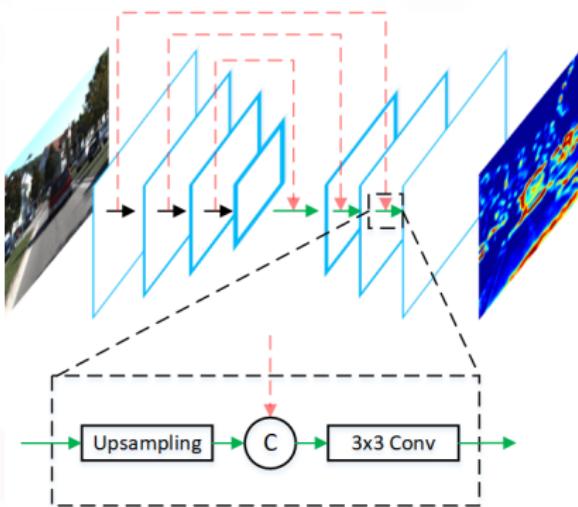


Figure: VGG16 architecture [4].

- Modified by resizing and reducing the layers.

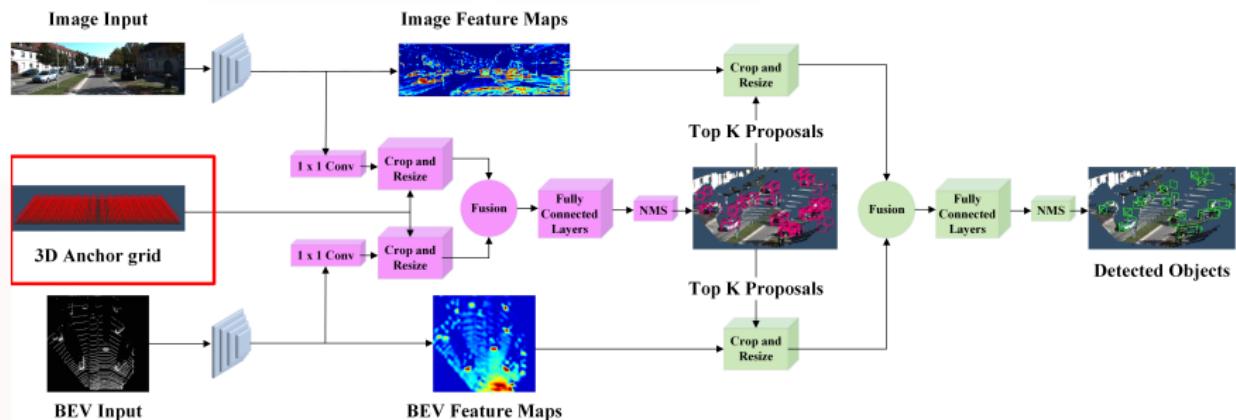
## Generating Feature Maps

- Feature Pyramid Feature Extractor



**Figure:** Feature maps are fused at every layer by an *upsampling* layer, followed by *concatenation*, and then mixing via a *convolutional* layer, resulting in a full resolution feature map.

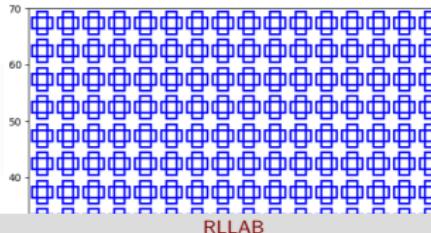
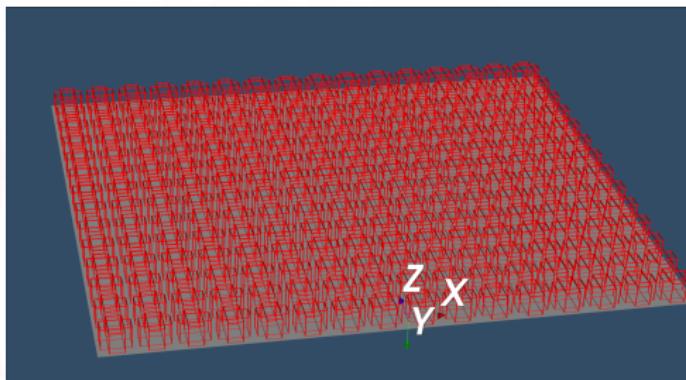
# RPN - Anchor Generation



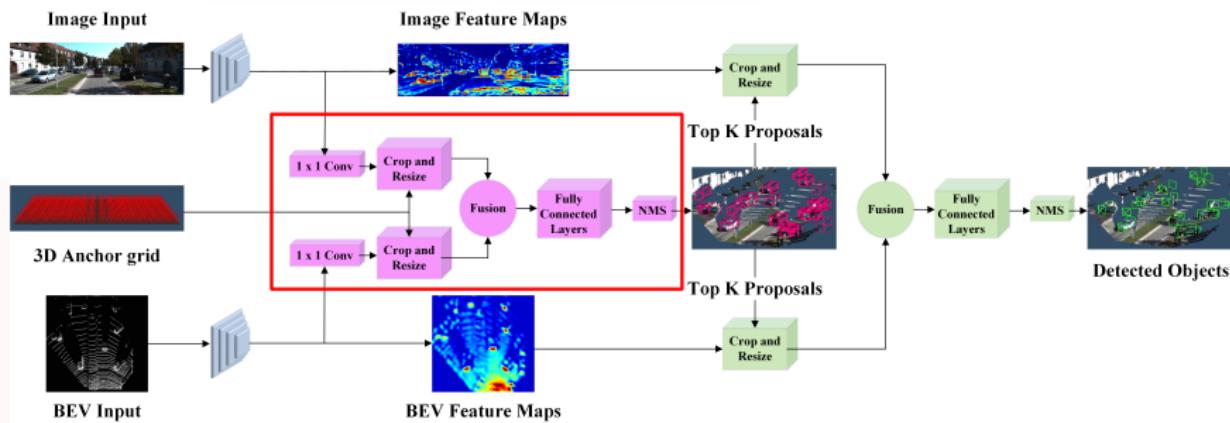
**Figure:** AVOD architectural diagram. The feature extractors are shown in **blue**, the region proposal network in **pink**, and the second stage detection network in **green**.

# RPN - Anchor Generation

- **Anchors:** Initial guesses on where objects are located.
  - Encoded as  $[x, y, z, l, w, h]$
- Take 3D box sizes from ground truth clusters.
- Generated on a grid on the ground plane, with  $0^\circ$  or  $90^\circ$  orientations.

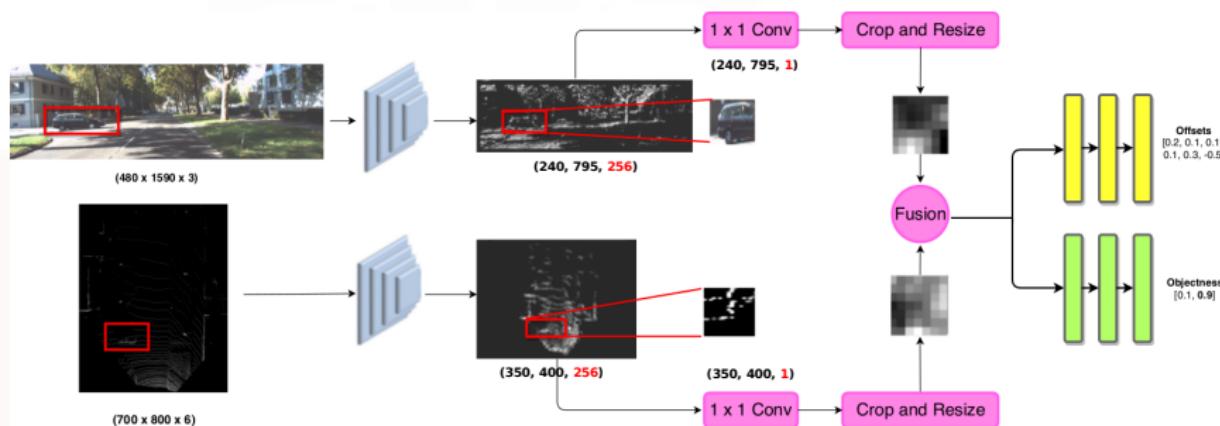


## Region Proposal Network



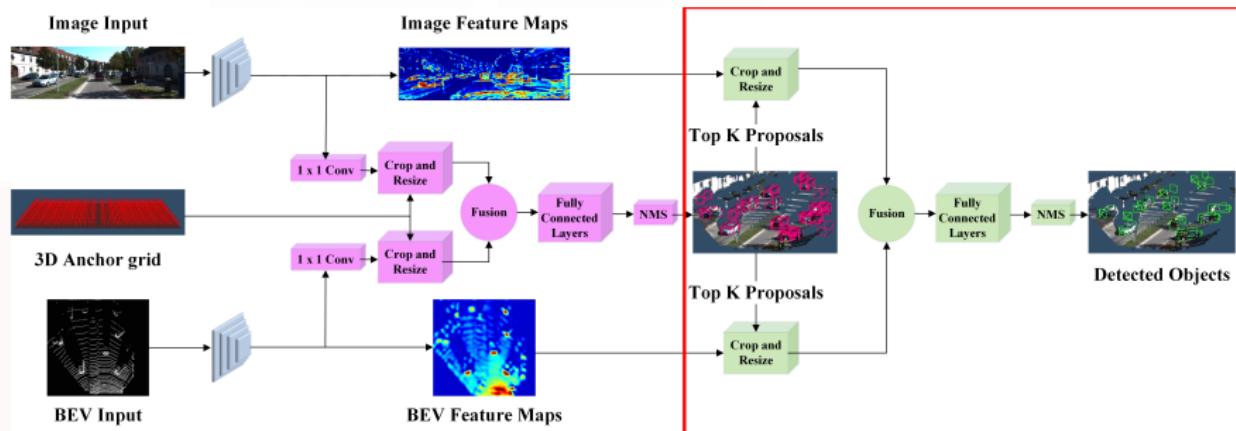
**Figure:** AVOD architectural diagram. The feature extractors are shown in **blue**, the region proposal network in **pink**, and the second stage detection network in **green**.

## Region Proposal Network



**Figure:** RPN fusion & proposal generation.

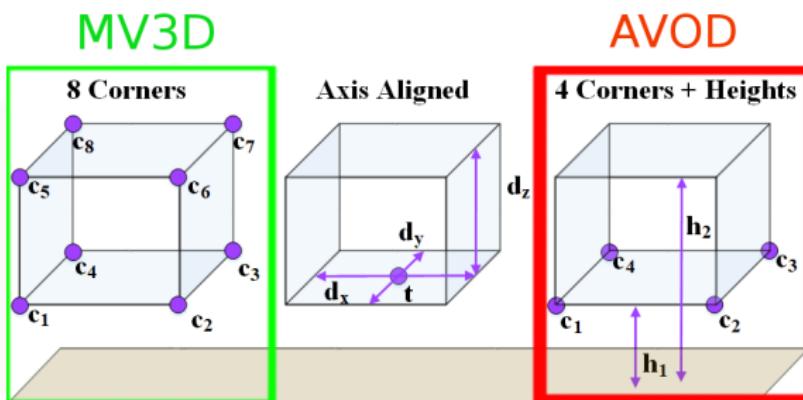
# Second Stage Detection Network



**Figure:** AVOD architectural diagram. The feature extractors are shown in **blue**, the region proposal network in **pink**, and the second stage detection network in **green**.

# Second Stage Detection Network

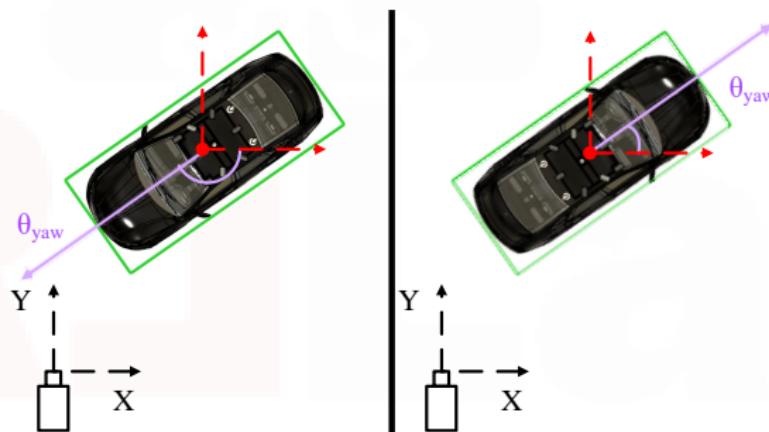
- 3D Bounding Box Encoding
  - Four corners and two heights.



**Figure:** A visual comparison between the 8 corner box encoding, the axis aligned box encoding and AVOD's 4 corner encoding.

# Explicit Orientation Vector Regression

- AVOD remedies this by computing:  $(x_{or}, y_{or}) = (\cos(\theta), \sin(\theta))$
- Handles angle wrapping as every  $\theta \in [-\pi, \pi]$  can be represented by a unique unit vector in the BEV space.



**Figure:** Object's bounding box does not change when the orientation (**purple**) is shifted by  $\pm\pi$  radians.

# Second Stage Detection Network

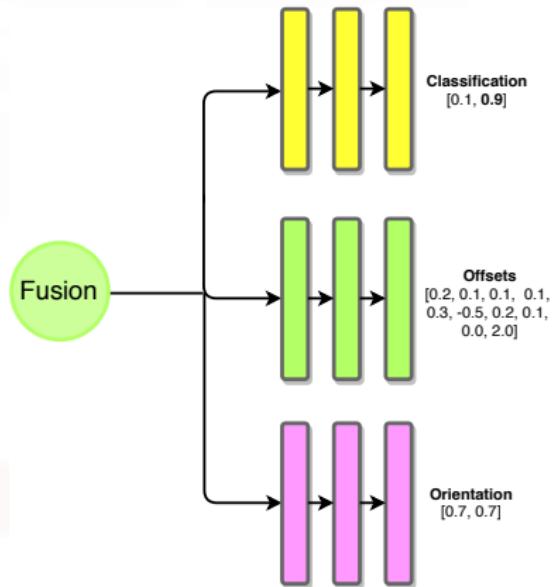


Figure: AVOD second stage task specific layers.

# Orientation Ambiguity

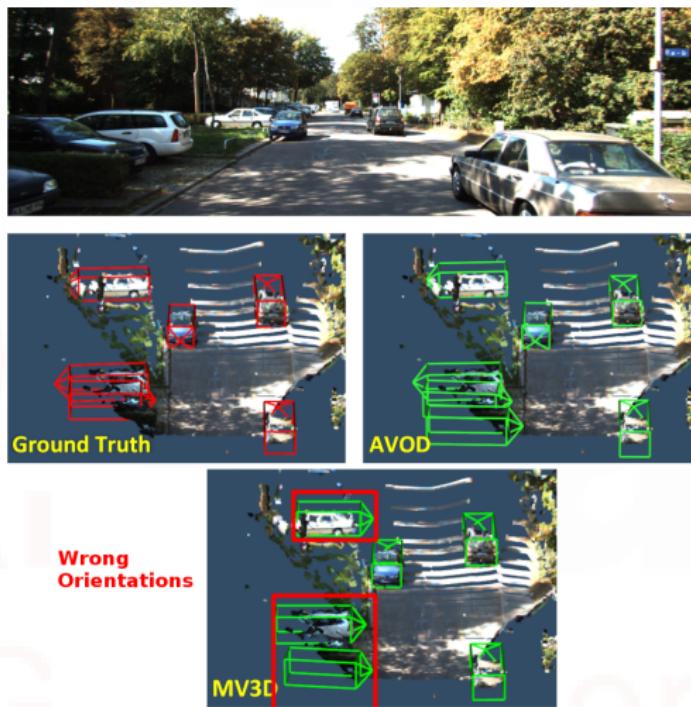
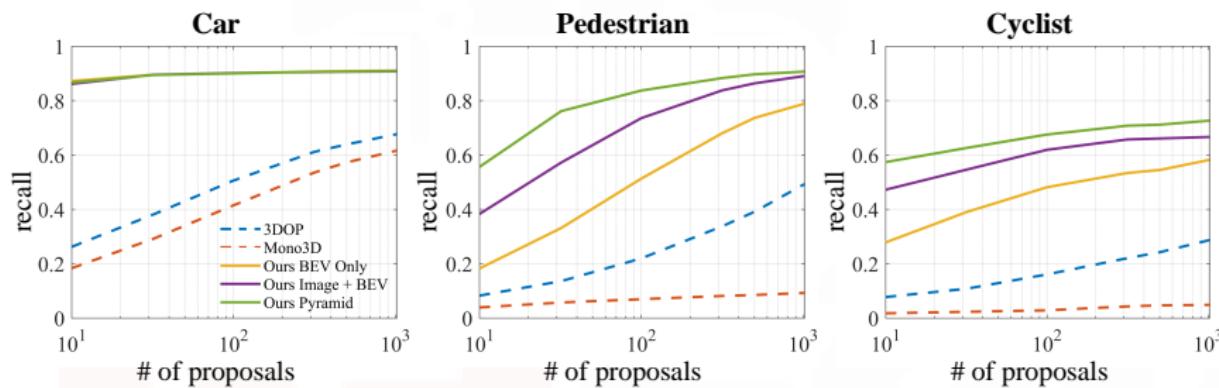


Figure: Qualitative comparison between MV3D and AVOD.

# RPN Input Variations



**Figure:** The recall vs number of proposals at a 3D IoU threshold of 0.5.

# AVOD - Results on Car Detection

Method	Moderate	Easy	Hard	Runtime
AVOD-FPN	<b>71.88</b>	<b>81.94</b>	<b>66.38</b>	0.1
F-PointNet	70.39	81.20	62.19	0.17
AVOD	65.78	73.59	58.38	<b>0.08</b>
VoxelNet	65.11	77.47	57.73	0.25
MV3D	62.35	71.09	55.12	0.36

**Table:** A comparison of the performance of AVOD with the state of the art 3D object detectors evaluated on KITTI's dataset. Results are generated by KITTI's evaluation server.

# AVOD - Results on Pedestrian Detection

Method	Moderate	Easy	Hard	Runtime
F-PointNet	<b>44.89</b>	<b>51.21</b>	40.23	0.17
AVOD-FPN	42.81	50.80	<b>40.88</b>	0.1
VoxelNet	33.69	39.48	31.51	0.25
AVOD	31.51	38.28	26.98	<b>0.08</b>

**Table:** A comparison of the performance of AVOD with the state of the art 3D object detectors evaluated on KITTI's dataset. Results are generated by KITTI's evaluation server.

# AVOD - Results on Cyclist Detection

Method	Moderate	Easy	Hard	Runtime
F-PointNet	<b>56.77</b>	<b>71.96</b>	<b>50.39</b>	0.17
AVOD-FPN	52.18	64.00	46.61	0.1
VoxelNet	48.36	61.22	44.37	0.25
AVOD	44.90	60.11	38.80	<b>0.08</b>

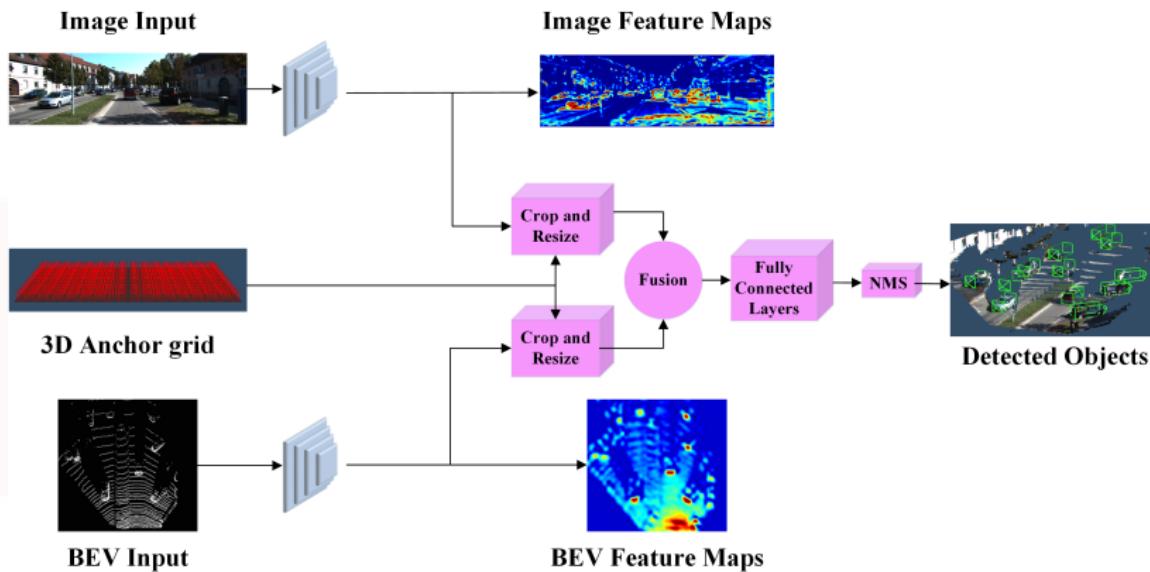
**Table:** A comparison of the performance of AVOD with the state of the art 3D object detectors evaluated on KITTI's dataset. Results are generated by KITTI's evaluation server.

# Motivation: Single Stage Detector

- The highest accuracy object detectors to date are based on two-stage detectors.
- Single stage object detection algorithms are *simple*:
  - Eliminate proposal generation and feature resampling stage.
  - Encapsulate all computation into a **single** stage.

McGill University

# AVOD-Single Stage Detector Architecture



## Class Imbalance



- Evaluate 10 – 100K candidate locations but only a few locations contain objects.
  - This imbalance causes two problems:
    - Training is *inefficient*.
    - Easy negatives are overwhelming.
  - Common solutions:
    - Hard negative mining.
    - More complex sampling technique.

# Focal Loss

- Designed to address class imbalance by down-weighting easy examples:
    - So that their contribution to the total loss is small even if their number is large.
  - Focuses training on *hard* examples.

## Focal Loss Continued



- To introduce the formulation of focal loss, we start with cross entropy (CE) loss for binary classification:

$$CE(p, y) = \begin{cases} -\log(p), & \text{if } y = 1 \\ -\log(1 - p), & \text{otherwise} \end{cases}$$

- In the above equation  $y \in \{\pm 1\}$  specifies the ground-truth class and  $p \in [0, 1]$  is the model's estimated probability for the class with label  $y = 1$ .

## Focal Loss Continued



- For notational convenience, we define  $p_t$ :

$$p_t = \begin{cases} p, & \text{if } y = 1 \\ 1 - p, & \text{otherwise} \end{cases}$$

- Then we can rewrite  $\text{CE}(p, y) = \text{CE}(p_t) = -\log(p_t)$ .

## Balanced Cross Entropy



- Weighting factor  $\alpha \in [0, 1]$ .
  - $\alpha$  for class 1, and  $1 - \alpha$  for class  $-1$ .

$$BCE(p_t) = -\alpha_t \log(p_t)$$

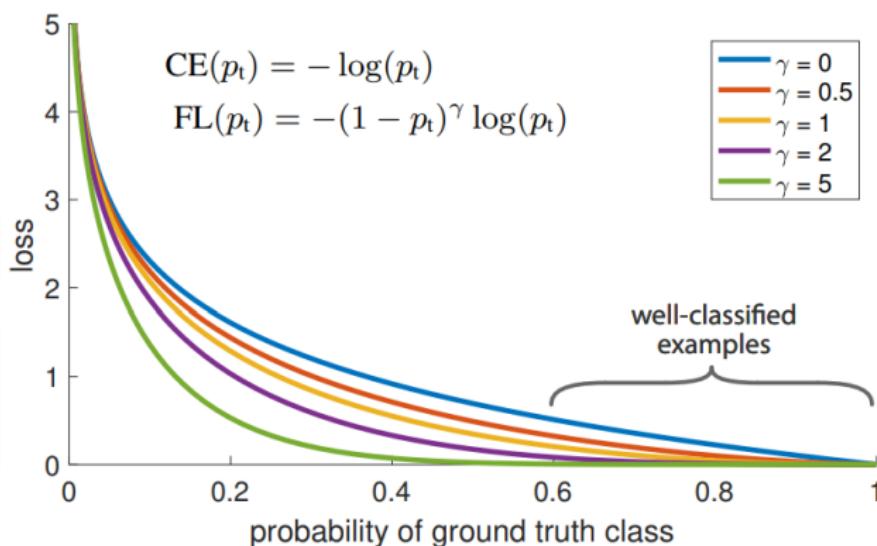
## Focal Loss Definition



- $\alpha$  balances the importance of **positive** and **negative** examples, it does not differentiate between **easy** and **hard** examples.
  - Focal loss uses a modulating factor  $(1 - p_t)^\gamma$  with the cross entropy loss, with a tunable *focusing* parameter  $\gamma \geq 0$ .

$$FL(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t)$$

## Focal Loss Plot



**Figure:** The focal loss visualized for several values of  $\gamma \in [0, 5]$  [7].

## AVOD-SSD Experiment Results



Architecture	Car		
	$AP_{3D}(0.7)$	$AHS_{3D}(0.7)$	Runtime speed (s)
AVOD	74.23	73.97	<b>0.08</b>
AVOD-FPN	<b>74.81</b>	<b>74.53</b>	0.1
AVOD-FPN(High Res BEV)	73.74	73.11	0.14
AVOD-SSD(FPN) <b>without</b> Focal Loss	61.52	60.32	0.09
AVOD-SSD(VGG)	56.11	55.89	0.2
AVOD-SSD(FPN)	73.93	73.32	0.09
AVOD-SSD(FPN + High Res BEV)	74.53	74.08	0.16

**Table:** Performance analysis of **AVOD**, **AVOD-FPN** and **AVOD-SSD** on the *validation* set and *moderate* difficulty.

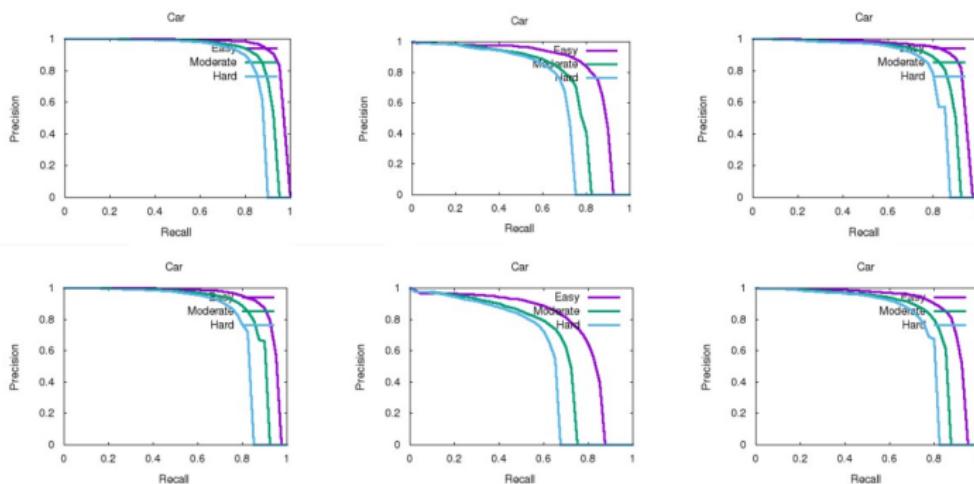
## AVOD-SSD Experiment Results



			$AP_{3D}$ (%)			$AP_{BEV}$ (%)			$AP_{2D}$ (%)		
Method	Runtime (s)	Class	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
AVOD-FPN	0.1	Car	<b>81.94</b>	<b>71.88</b>	<b>66.38</b>	<b>88.53</b>	83.79	<b>77.90</b>	<b>89.99</b>	87.44	80.05
AVOD	<b>0.08</b>		73.59	65.78	58.38	86.80	<b>85.44</b>	77.73	89.73	<b>88.08</b>	<b>80.14</b>
AVOD-SSD	0.09		73.64	63.87	56.90	86.14	77.66	75.68	88.94	85.71	78.05

**Table:** Performance analysis of **AVOD**, **AVOD-FPN** and **AVOD-SSD** on the KITTI's evaluation server.

## Recall and Precision of AVOD vs AVOD-SSD



**Figure:** Comparison of Precision-Recall curve of AVOD-FPN(top) vs AVOD-SSD(bottom) for 2D, 3D and BEV detection respectively, on the *test* set.

## AVOD Limitations



- 2D representation of 3D data
    - Raw 3D data has much richer information where such information is being discarded.
  - Accurate estimated ground-plane assumption.
  - Objects appear within certain distance above the ground-plane.

## Future Work



- Speed Optimization
  - Faster 3D Single Stage Detector
  - Processing Temporal Information
    - Combined pipeline for object detection and tracking where tracking information can also guide object detection and vice versa.

## References



- Kitti 3D Object Detection Benchmark. Access: [http://www.cvlibs.net/datasets/kitti/eval\\_object.php?obj\\_benchmark=3d](http://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d).
  - Artificial Intelligence GitBook. Leonardo Araujo dos Santos. Access: [https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/object\\_localization\\_and\\_detection.html](https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/object_localization_and_detection.html).
  - Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Ren, Shaoqing and He, Kaiming and Girshick, Ross and Sun, Jian. Access: <https://arxiv.org/abs/1506.01497>.
  - VGG16 Architecture. Heuritech. Access: <https://blog.heuritech.com/2016/02/29/a-brief-report-of-the-heuritech-deep-learning-meetup-5/>.
  - Multi-view 3d object detection network for autonomous driving. Chen, Xiaozhi and Ma, Huimin and Wan, Ji and Li, Bo and Xia, Tian. Access: <https://arxiv.org/abs/1611.07759>.
  - Frustum PointNets for 3D Object Detection from RGB-D Data. Qi, Charles R and Liu, Wei and Wu, Chenxia and Su, Hao and Guibas, Leonidas J. Access: <https://arxiv.org/abs/1711.08488>.
  - Focal loss for dense object detection. Lin, Tsung-Yi and Goyal, Priya and Girshick, Ross and He, Kaiming and Dollár, Piotr. Access: [arXivpreprintarXiv:1708.02002](https://arxiv.org/abs/1708.02002).
  - You only look once: Unified, real-time object detection. Redmon, Joseph and Divvala, Santosh and Girshick, Ross and Farhadi, Ali. Proceedings of the IEEE conference on computer vision and pattern recognition

Thank you!  
Questions?