PointPillars: Fast Encoders for Object Detection from Point Clouds

- Alex H. Lang, Sourabh Vora, Holger Caesar, Luming Zhou, Jiong Yang, Oscar Beijbom. Fast encoders for object detection from point clouds. In CVPR, 2019

Sushruth Nagesh
ECE Department, UCSD
Deploying autonomous vehicles (AV's) in urban areas is technologically challenging. Need to detect and track moving objects like vehicles, pedestrians and cyclists for safe operation of AV's. LIDAR, RADAR and image sensors are some of the few sensors used to achieve this objective.
Objective

- Robust and real time object detection in point clouds
- Propose a learnable encoder - PointPillars to leverage the information represented by the point cloud
- Build an effective downstream object detection pipeline
Existing methodologies

- LIDAR based methods
  - Bottom up approach – Build occupancy grid, extract connected components or train an SVM classifier on top.
  - Deep learning based approaches – Projecting point clouds to 2D images and use conventional CNN based pipeline. MLP based (PointNet). 3D convolution based (VoxelNet, SECOND).

- Fusion based methods (LIDAR/Image/RADAR)
  - AVOD, MV3D – Fuse the lidar features with image features to create a multimodal detector
PointPillars

- A method for object detection in 3D that enables end-to-end learning with only 2D convolutional layers.
- PointPillars uses a novel encoder that learn features on pillars (vertical columns) of the point cloud to predict 3D oriented boxes for objects.
- Accurate and fast object detection based on the proposed encoder.
Point Pillars Network

- **Point Cloud to Pseudo Image**
  - Point $l$ in point cloud – $x, y, z, r$
  - Point cloud is discretized into evenly spaced grid in x-y plane. Evenly spaced grid - Pillars
  - Augment the point with $x_c, y_c, z_c, x_p$ and $y_p$ so that point $l$ has $D=9$ dimensions
  - Subscript $c$ denotes the distance from geometric center of all the points in the pillar. Subscript $p$ denotes offset from pillar center
  - A dense tensor of size $(D, P, N)$ is created. $P$ – Number of pillars, $N$ – number of points per pillar
  - PointNet is applied to produce a tensor $(C, P)$ which is reshaped to $(C, H, W)$. $H$, $W$ are the height and width of the pseudo image
PointPillars Network

- 2D convolutional backbone
  - Objective is to transform the pseudo image to a higher-level representation
  - Top down network to produce features at smaller spatial resolution. Characterized by series of blocks (S, L, F) where S – Stride, L – 3x3 2D Conv layer and F - output channels.
  - Upsampling and Concatenation - Upsampling using transposed 2D conv layers. Concatenation of output features at different levels.
In general there are two types of detection mechanisms. Two stage architectures (R-CNN, Fast R-CNN and Faster R-CNN). Single stage architectures (SSD and YOLO).

Two stage architectures have one stage for region proposal, another for classification.

Single stage architectures – region proposal and classification in single shot.

Single stage architectures are fast and accurate. So, Single Shot detector (SSD) is used in PointPillars.
Loss functions

- Ground Truth boxes and anchors are defined by $b \in (x, y, z, w, l, h, \theta)$.
- Localization or regression loss is smooth L1 loss defined by $b$.
  \[ L_{loc} = \sum_b \text{SmoothL1}(\Delta b) \]
- Focal loss is used as object classification loss.
  \[ L_{cls} = -\alpha (1 - p^a)^\gamma \log(p^a) \]
- Softmax classification loss $L_{dir}$ for discretized directions
- Total loss is weighted sum of all the three losses
Settings

- Two networks. One for car detection and another for pedestrian and cyclist detection.
- Two IOU thresholds for positive and negative anchor box matches.
- Different anchor box width, height and length for cars, pedestrians and cyclists.
- Data augmentation by creating a lookup table of the ground truth 3D boxes for all classes and the associated point clouds. Randomly concatenating to the current training point cloud. Collision test performed.
- Global and local augmentation (Translation, rotation).
Results

- KITTI evaluation metrics including bird’s eye view (BEV), 3D and Average orientation similarity (AOS).
- Outperforms all LiDAR based methods with respect to mean average precision (mAP) on car, cyclist and pedestrian categories (BEV and 3D).
- Outperforms all fusion based methods on car and cyclist categories (BEV and 3D).
- AOS is done by projection of 3D boxes to image and performing 2D detection matching, and then assessing the orientation of these matches.
- Good performance on AOS but still image-based methods are SOTA.
Results Examples
Realtime Inference

- Runtime measured on Intel i7 and Nvidia 1080ti GPU
- Total inference time for a given point cloud is 16.2 ms (~ 45 Hz) of which encoder takes 1.3ms.
- Considerably fast encoder compared to VoxelNet Encoder (190ms) and SECOND encoder (50ms)
- Can achieve higher speed (~ 105 Hz) by increasing the bin size. But tradeoff with accuracy
Advantages

- By learning features instead of relying on fixed encoders, PointPillars can leverage the full information represented by the point cloud.

- Finally, pillars are highly efficient because all key operations can be formulated as 2D convolutions which are extremely efficient to compute on a GPU.

- Minimal box augmentation compared to VoxelNet and SECOND.

- No hand-tuning to use different point cloud configurations. For example, it can easily incorporate multiple lidar scans.
Limitations

- Only ~10% of the total point cloud in KITTI is utilized. i.e only LIDAR points that can be projected onto a front image is used.
- An operational AV needs to view the full environment and process the complete point cloud.
- Also measurements are taken on a Desktop computer. However AV’s operate on embedded CPU and compute units with lesser throughputs.
- Pedestrians and cyclists are commonly misclassified as each other. Additionally, pedestrians are easily confused with narrow vertical features of the environment such as poles or tree trunks.
Failure Examples
Future Improvements

- Fusion of LIDAR, RADAR and Image point clouds. (Helps to overcome individual limitations). This would give better accuracy.

- Reduce the inference time to achieve real time operation. Can try to use MobileNet or CornerNet instead of SSD

- Develop models for semantic segmentation and instance segmentation based on PointPillars
Key Takeaway

PointPillars - Learnable encoder based on vertical column partitions (Pillars)
Can do end to end learning
Can use this architecture with different downstream detection pipelines
Questions....
Thank you