First Paper: PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection.

Name: Mohamed Said Ibrahem **ID: 57**

Usage: **For accurate 3D object detection from point clouds.**

Note that: **A point cloud:** is a set of data points in space. Point clouds are generally produced by 3D scanners, which measure many points on the external surfaces of objects around them.

The method integrates both Convolutional Neural Network (CNN) and PointNet-based networks.

Why?
Extensive experiments on both the KITTI dataset and the Waymo Open dataset show that our proposed PV-RCNN surpasses state-of-the-art 3D detection methods with remarkable margins by using only point clouds.

RoI-grid feature points encode much richer context information for accurately estimating object confidences and locations

ROI : **region of interest** are samples within a data set identified for a particular purpose. The concept of a ROI is commonly used in many application areas. For example, in medical imaging, the boundaries of a tumor may be defined on an image or in a volume, for the purpose of measuring its size.

What is the importance of 3D object detection?

As it is used in wide applications in various fields such as autonomous driving and robotics.

What is Our Purpose?

to achieve high performance 3D object detection by designing novel point-voxel integrated networks and to design novel point-voxel integrated networks to learn better 3D features from irregular point clouds.

Most existing 3D detection methods could be classified into two categories in terms of point cloud representations:
1- grid-based methods.
2- point-based methods.

●The grid-based methods generally transform the irregular point clouds to regular representations such as 3D voxels or 2D bird-view.

Which could be efficiently processed by 3D or 2D Convolutional Neural Networks (CNN) to learn point features for 3D detection.

●The point-based methods directly extract discriminative features from raw point clouds for 3D detection.

PV-RCNN boosts the 3D detection performance by incorporating the advantages from both the Point-based and Voxel-based feature learning methods.

What is the main challenge?
The main challenge would be how to effectively combine the two types of feature learning schemes, specifically the 3D voxel CNN with sparse convolutions and the PointNet-based set abstraction into a unified framework.
There are two solutions for this challenge:

1- uniformly sampling several grid points within each 3D proposal, and adopting the set abstraction to aggregate 3D voxel-wise features surrounding these grid points for proposal refinement.
Disadvantages: highly memory-intensive since both the number of voxels and the number of grid points could be quite large to achieve satisfactory performance.

To better integrate these two types of point cloud feature learning networks:
We propose a two-step strategy with the first **voxel-to-keypoint** scene encoding step and the second **keypoint-to-grid** RoI(region of interest) feature abstraction step.

To mitigate the above mentioned issue of requiring too many voxels for encoding the whole scene, a small set of key points are selected by the furthest point sampling (FPS) ابعد نقطة to summarize the overall 3D information from the voxel-wise features

all grid points' aggregated features can be jointly used for the succeeding proposal refinement.

PV-RCNN effectively takes advantage of both point-based and voxel-based networks to encode discriminative features at each box proposal for accurate confidence prediction and fine-grained box refinement.

The proposal can be summarized into four folds:
1- Propose PV-RCNN framework which effectively takes advantages of both the voxel-based and point-based methods for 3D point-cloud feature learning, leading to improved performance of 3D object detection with manageable memory consumption.
2- Propose the voxel-to-keypoint scene encoding scheme, which encodes multiscale voxel features of the whole scene to a small set of

keypoints by the voxel set abstraction layer
3- Propose a multi-scale RoI feature abstraction layer for grid points in each proposal, which aggregates richer context information from the scene with multiple receptive fields for accurate box refinement and confidence prediction.
4- proposed method PV-RCNN outperforms all previous methods with remarkable margins and ranks 1 st on the highly competitive KITTI 3D detection benchmark.

PV-RCNN for Point Cloud Object Detection

PointVoxel-RCNN (PV-RCNN), is a two-stage 3D detection framework aiming at more accurate 3D object detection from point clouds.

3D detection approaches are based on either 3D voxel CNN with sparse convolution or PointNet-based networks as the backbone.

3D voxel CNNs with sparse convolution are more efficient and are able to generate high-quality 3D object proposals.

PointNet-based methods can capture more accurate contextual information with flexible receptive fields.

PV-RCNN deeply integrates the advantages of two types of networks.

PV-RCNN consists of a 3D voxel CNN with sparse convolution as the backbone for efficient feature encoding and proposal generation. Given each 3D object proposal, there are two novel operations to effectively pool its corresponding features from the scene:
1- voxel-to-keypoint scene encoding: summarizes all the voxels of the overall scene feature volumes into a small number of feature keypoints.
2- point-to-grid RoI feature abstraction: effectively aggregates the scene keypoint features to RoI grids for proposal confidence prediction and location refinement.

3D Voxel CNN for Efficient Feature Encoding and Proposal Generation.

Voxel CNN with 3D sparse convolution is a popular choice by state-of-the-art 3D detectors for efficiently converting the point clouds into sparse 3D feature volumes. Because of its high efficiency and accuracy, we adopt it as the backbone of our framework for feature encoding and 3D proposal generation.

Input points P are first divided into small voxels with spatial resolution of L × W × H, where the features of the non-empty voxels are directly calculated as the mean of pointwise features of all inside points.

The network utilizes a series of 3 × 3 × 3 3D sparse convolution to gradually convert the point clouds into feature volumes with 1×, 2×, 4×, 8× downsampled sizes.

Such sparse feature volumes at each level could be viewed as a set of voxel-wise feature vectors. 3D proposal generation. By converting the encoded 8×downsampled 3D feature volumes into 2D bird-view feature maps, high-quality 3D proposals are generated following the anchor-based approaches. Specifically, we stack the 3D feature volume along the Z axis to obtain the L/8 x W/8 bird-view feature maps. Each class has 2 × L/8 × W/8 3D anchor boxes which adopt the average 3D object sizes of this class, and two anchors of 0, 90 orientations are evaluated for each pixel of the bird-view feature maps. As shown in Table 4, the adopted 3D voxel CNN backbone with anchor-based scheme achieves higher recall performance than the PointNet-based approaches.

A naive solution of using the set abstraction operation for pooling the scene feature voxels would be directly aggregating the multi-scale feature volume in a scene to the RoI grids. However, this intuitive strategy simply occupies much memory and is inefficient to be used in practice.

A common scene from the KITTI dataset might result in 18, 000 voxels in the 4× downsampled feature volumes. If one uses 100 box proposals for each scene and each box proposal has 3 × 3 × 3 grids. The 2, 700 × 18, 000 pairwise distances and feature aggregations cannot be efficiently computed, even after distance thresholding.

Voxel-to-keypoint Scene Encoding via Voxel Set Abstraction

The framework first aggregates the voxels at the multiple neural layers representing the entire scene into a small number of keypoints, which serve as a bridge between the 3D voxel CNN feature encoder and the proposal refinement network.

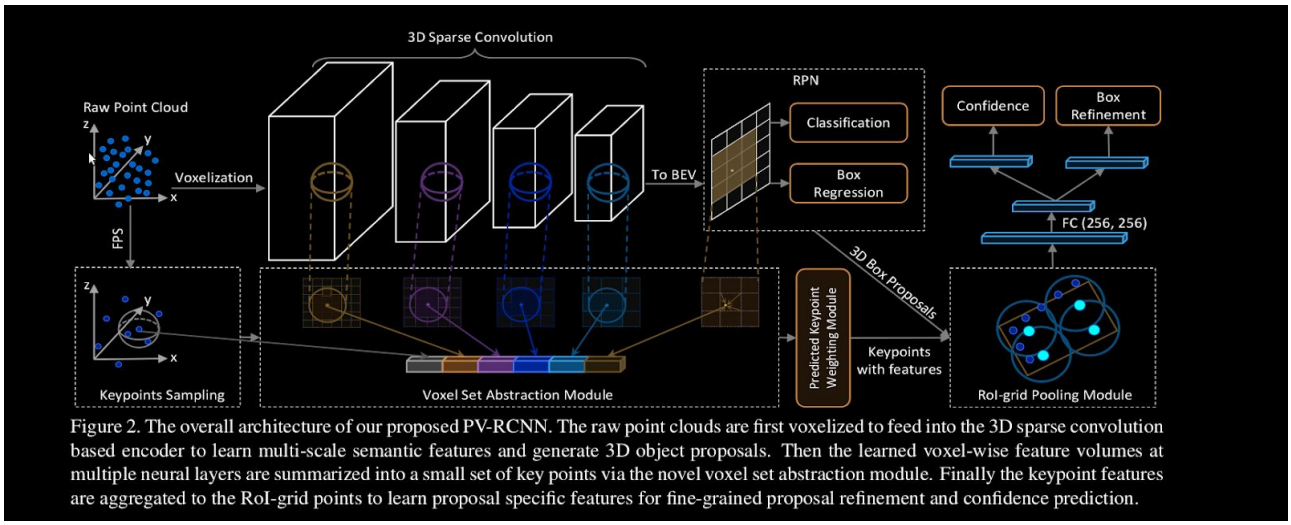


Figure 2. The overall architecture of our proposed PV-RCNN. The raw point clouds are first voxelized to feed into the 3D sparse convolution based encoder to learn multi-scale semantic features and generate 3D object proposals. Then the learned voxel-wise feature volumes at multiple neural layers are summarized into a small set of key points via the novel voxel set abstraction module. Finally the keypoint features are aggregated to the RoI-grid points to learn proposal specific features for fine-grained proposal refinement and confidence prediction.

Key Points Sampling. adopt the Furthest Point-Sampling (FPS) algorithm to sample a small number of n keypoints K = {p 1 , · · · , p n } from the point clouds P, where n = 2, 048 for the KITTI dataset and n = 4, 096, Such a strategy encourages that the keypoints are uniformly distributed around non-empty voxels and can be representative to the overall scene.

Voxel Set Abstraction Module. the Voxel Set Abstraction (VSA) module to encode the multi-scale semantic features from the 3D CNN feature volumes to the keypoints. The set abstraction operation proposed by is adopted for the aggregation of voxel-wise feature volumes.

Denote $\mathcal{F}^{(l_k)} = \{f_1^{(l_k)}, \cdots, f_{N_k}^{(l_k)}\}$ as the set of voxel-wise feature vectors in the k-th level of 3D voxel CNN.

And $\mathcal{V}^{(l_k)} = \{v_1^{(l_k)}, \cdots, v_{N_k}^{(l_k)}\}$ as their 3D coordinates calculated by the voxel indices and actual voxel sizes of the k-th level

where N_k is the number of non-empty voxels in the k-th level

Illustration of Predicted Keypoint Weighting module.

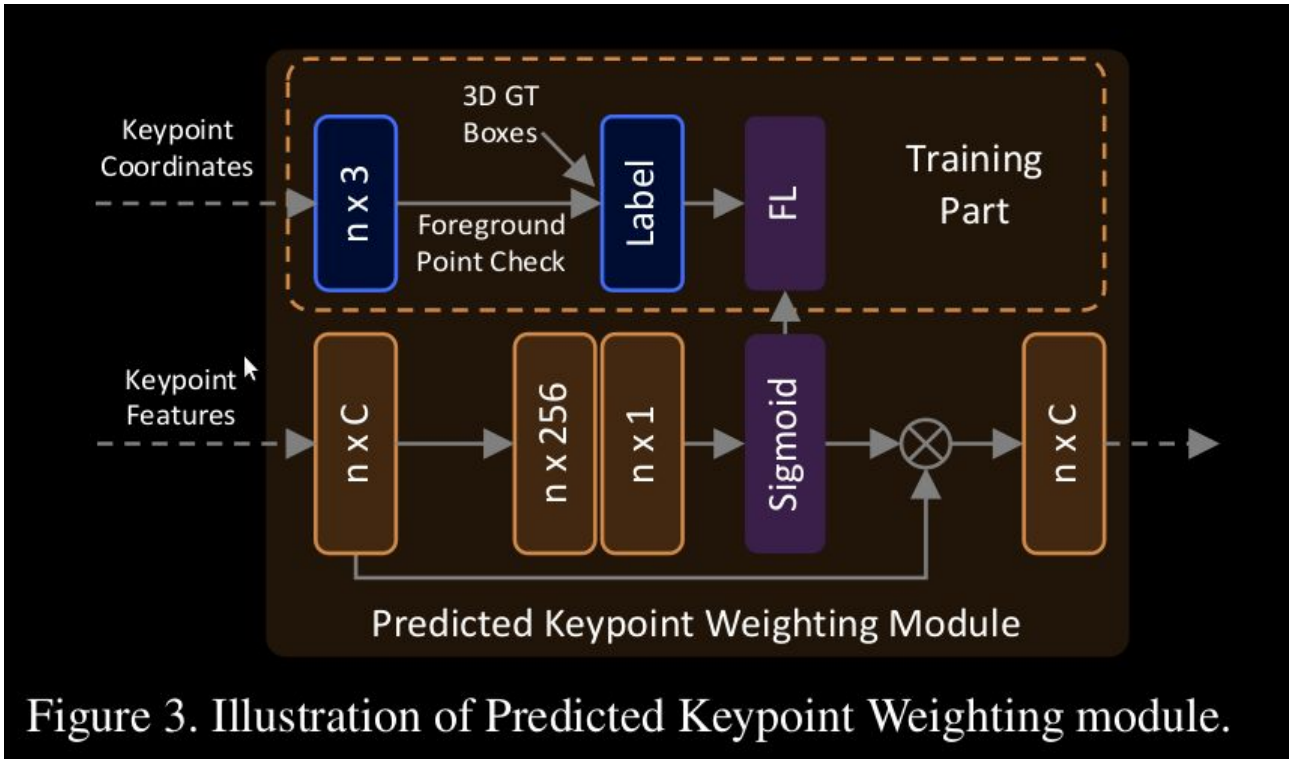


Figure 3. Illustration of Predicted Keypoint Weighting module.

Illustration of RoI-grid pooling module. Rich context information of each 3D RoI is aggregated by the set abstraction operation with multiple receptive fields.

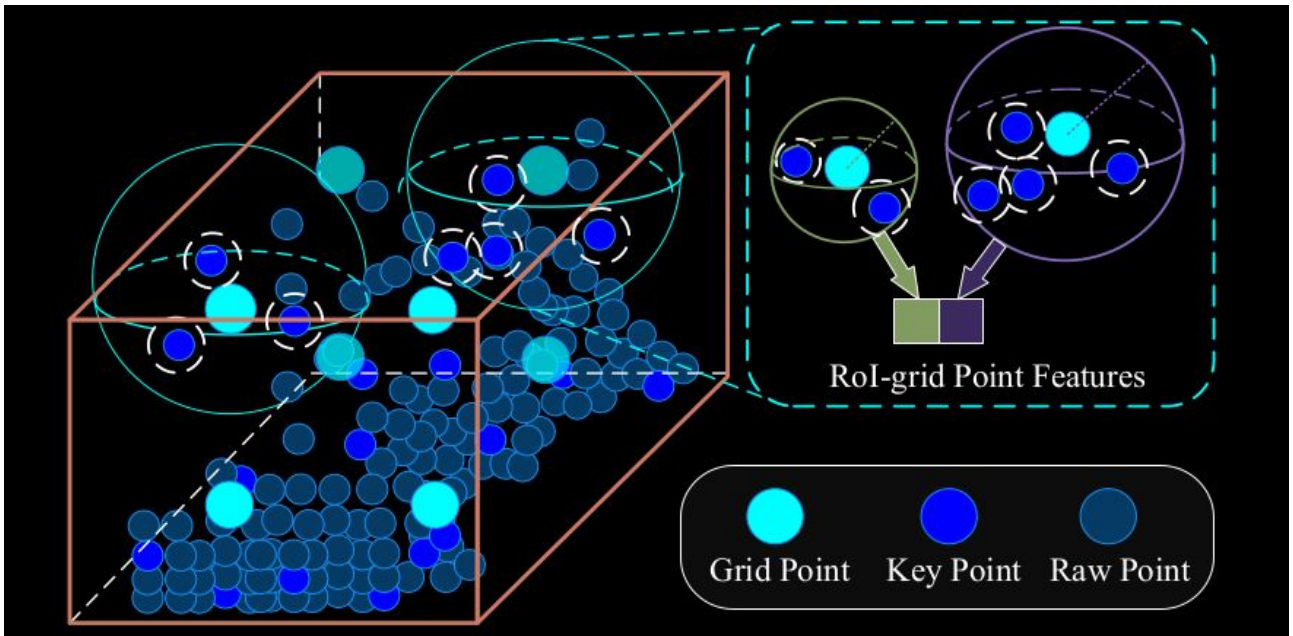


Figure 4. Illustration of RoI-grid pooling module. Rich context information of each 3D RoI is aggregated by the set abstraction operation with multiple receptive fields.

Implementation details of our PV-RCNN framework

Dataset:

KITTI Dataset is one of the most popular datasets of 3D detection for autonomous driving. There are 7, 481 training samples and 7, 518 test samples, where the training samples are generally divided into the train split (3, 712 samples) and the val split (3, 769 samples). We compare PV-RCNN with state-of-the-art methods on both the val split and the test split on the online leaderboard.

Network Architecture:

3D voxel CNN has four levels with feature dimensions 16, 32, 64, 64, respectively as shown in this figure.

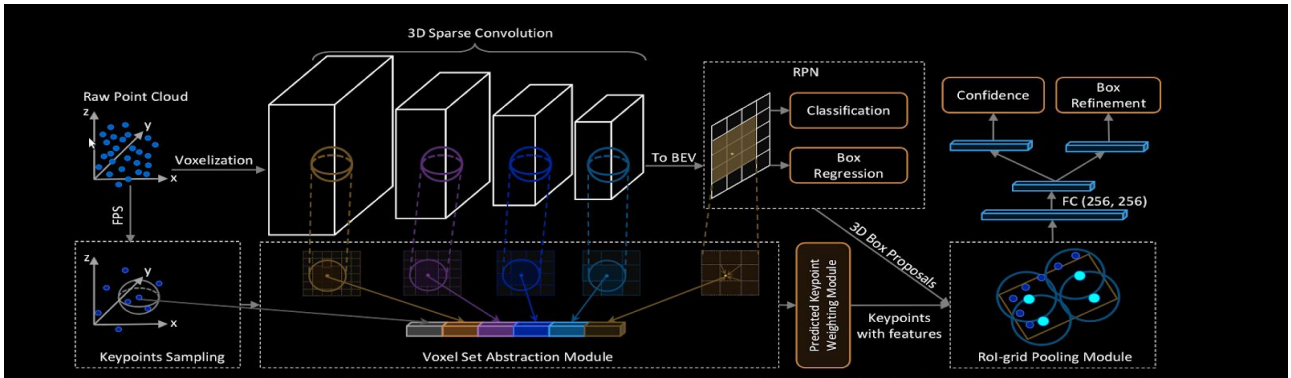


Figure 2. The overall architecture of our proposed PV-RCNN. The raw point clouds are first voxelized to feed into the 3D sparse convolution based encoder to learn multi-scale semantic features and generate 3D object proposals. Then the learned voxel-wise feature volumes at multiple neural layers are summarized into a small set of key points via the novel voxel set abstraction module. Finally the keypoint features are aggregated to the RoI-grid points to learn proposal specific features for fine-grained proposal refinement and confidence prediction.

Performance comparison:

Method	Reference	Modality	Car - 3D Detection			Car - BEV Detection			Cyclist - 3D Detection			Cyclist - BEV Detection		
			Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
MV3D [1]	CVPR 2017	RGB + LiDAR	74.97	63.63	54.00	86.62	78.93	69.80	-	-	-	-	-	-
ContFuse [17]	ECCV 2018	RGB + LiDAR	83.68	68.78	61.67	94.07	85.35	75.88	-	-	-	-	-	-
AVOD-FPN [11]	IROS 2018	RGB + LiDAR	83.07	71.76	65.73	90.99	84.82	79.62	63.76	50.55	44.93	69.39	57.12	51.09
F-PointNet [22]	CVPR 2018	RGB + LiDAR	82.19	69.79	60.59	91.17	84.67	74.77	72.27	56.12	49.01	77.26	61.37	53.78
UberATG-MMF [16]	CVPR 2019	RGB + LiDAR	88.40	77.43	70.22	93.67	88.21	81.99	-	-	-	-	-	-
SECOND [34]	Sensors 2018	LiDAR only	83.34	72.55	65.82	89.39	83.77	78.59	71.33	52.08	45.83	76.50	56.05	49.45
PointPillars [12]	CVPR 2019	LiDAR only	82.58	74.31	68.99	90.07	86.56	82.81	77.10	58.65	51.92	79.90	62.73	55.58
PointRCNN [25]	CVPR 2019	LiDAR only	86.96	75.64	70.70	92.13	87.39	82.72	74.96	58.82	52.53	**82.56**	67.24	60.28
3D IoU Loss [39]	3DV 2019	LiDAR only	86.16	76.50	71.39	91.36	86.22	81.20	-	-	-	-	-	-
Fast Point R-CNN [2]	ICCV 2019	LiDAR only	85.29	77.40	70.24	90.87	87.84	80.52	-	-	-	-	-	-
STD [37]	ICCV 2019	LiDAR only	87.95	79.71	75.09	94.74	89.19	**86.42**	**78.69**	61.59	55.30	81.36	67.23	59.35
Patches [13]	Arxiv 2019	LiDAR only	88.67	77.20	71.82	92.72	88.39	83.19	-	-	-	-	-	-
Part-A^2 [26]	Arxiv 2019	LiDAR only	87.81	78.49	73.51	91.70	87.79	84.61	-	-	-	-	-	-
PV-RCNN (Ours)	-	LiDAR only	**90.25**	**81.43**	**76.82**	**94.98**	**90.65**	86.14	78.60	**63.71**	**57.65**	82.49	**68.89**	**62.41**
Improvement	-	-	*+1.58*	*+1.72*	*+1.73*	*+0.24*	*+1.46*	*-0.28*	*-0.06*	*+2.12*	*+2.35*	*-0.07*	*+1.65*	*+2.13*

Table 1. Performance comparison on the KITTI *test* set. The results are evaluated by the mean Average Precision with 40 recall positions.

Method	Reference	Modality	3D mAP
MV3D [1]	CVPR 2017	RGB + LiDAR	62.68
ContFuse [17]	ECCV 2018	RGB + LiDAR	73.25
AVOD-FPN [11]	IROS 2018	RGB + LiDAR	74.44
F-PointNet [22]	CVPR 2018	RGB + LiDAR	70.92
VoxelNet [41]	CVPR 2018	LiDAR only	65.46
SECOND [34]	Sensors 2018	LiDAR only	76.48
PointRCNN [25]	CVPR 2019	LiDAR only	78.63
Fast Point R-CNN [2]	ICCV 2019	LiDAR only	79.00
STD [37]	ICCV 2019	LiDAR only	79.80
PV-RCNN (Ours)	-	LiDAR only	**83.90**

Table 2. Performance comparison on the moderate level car class of KITTI *val* split with mAP calculated by 11 recall positions.

IoU Thresh.	3D mAP			BEV mAP		
	Easy	Moderate	Hard	Easy	Moderate	Hard
0.7	92.57	84.83	82.69	95.76	91.11	88.93

Table 3. Performance on the KITTI *val* split set with mAP calculated by 40 recall positions for car class.

Method	PointRCNN [25]	STD [37]	PV-RCNN (Ours)
Recall (IoU=0.7)	74.8	76.8	85.5

Table 4. Recall of different proposal generation networks on the car class at moderate difficulty level of the KITTI *val* split set.

3D Detection on the KITTI Dataset.

To evaluate the proposed model's performance on the KITTI val split, we train our model on the train set and report the results on the val set. To conduct evaluation on the test set with the KITTI official test server, the model is trained with 80% of all available train+val data and the remaining 20% data is used for validation.

Evaluation Metric. All results are evaluated by the mean average precision with a rotated IoU threshold 0.7 for cars and 0.5 for cyclists. The mean average precisions on the test set are calculated with 40 recall positions on the official KITTI test server. The results on the val set are calculated with 11 recall positions to compare with the results by the previous works.

Second Paper: Point-Voxel CNN for Efficient 3D Deep Learning.

3D deep learning has received increased attention thanks to its wide applications: e.g., AR/VR and autonomous driving These applications need to interact with people in real time and therefore require low latency. However, edge devices (such as mobile phones and VR headsets) are tightly constrained by hardware resources and battery. Therefore, it is important to design efficient and fast 3D deep learning models for real-time applications on the edge.

*3D data usually comes in the format of point clouds, Collected by the LiDAR sensors, researchers rasterize the point cloud into voxel grids and process them using 3D volumetric convolutions.

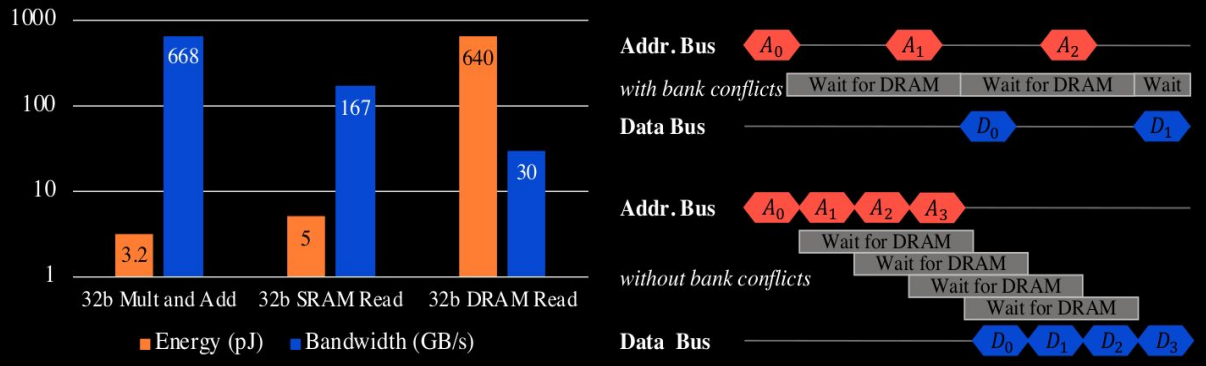
There will be information loss during voxelization: multiple points will be merged together if they lie in the same grid.

Therefore, a high-resolution representation is needed to preserve the fine details in the input data. However, the computational cost and memory requirement both increase cubically with voxel resolution.

As for point-based networks, up to 80% of the time is wasted on structuring the sparse data which have rather poor memory locality, not on the actual feature extraction. In this paper, we propose PVCNN that represents the 3D input data in points to reduce the memory consumption, while performing the convolutions in voxels to reduce the irregular, sparse data access and improve the locality.

PVCNN model is both memory and computation efficient.

Note That: Efficient 3D models should reduce memory footprint and avoid random memory accesses.



(a) Off-chip DRAM accesses take two orders of magnitude more energy than arithmetic operations (640pJ *vs.* 3pJ [10]), while the bandwidth is two orders of magnitude less (30GB/s *vs.* 668GB/s [16]). Efficient 3D deep learning should **reduce the memory footprint**, which is the bottleneck of conventional *voxel-based* methods.

(b) Random memory access is inefficient since it cannot take advantage of the DRAM burst and will cause bank conflicts [28], while contiguous memory access does not suffer from the above issue. Efficient 3D deep learning should **avoid random memory accesses**, which is the bottleneck of conventional *point-based* methods.

Figure 1: Efficient 3D models should reduce memory footprint and avoid random memory accesses.

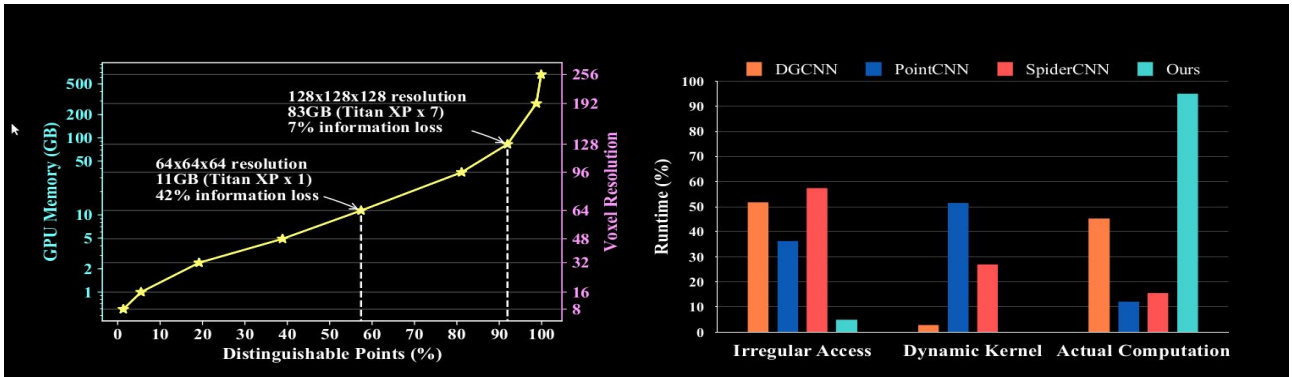
Most point-based models mimic the 3D volumetric convolution: they extract the feature of each point by aggregating its neighboring features. However, neighbors are not stored contiguously in the point representation; therefore, indexing them requires the costly nearest neighbor search.

To trade space for time, previous methods replicate the entire point cloud for each center point in the nearest neighbor search, and the memory cost will then be O(n 2), where n is the number of input points. Another overhead is introduced by the dynamic kernel computation. Since the relative positions of neighbors are not fixed, these point-based models have to generate the convolution kernels dynamically based on different offsets.

Designing efficient 3D neural network models needs to take the hardware into account. Compared with arithmetic operations, memory operations are particularly expensive: they consume two orders of magnitude higher energy, having two orders of magnitude lower bandwidth (Figure 1a). Another aspect is the memory access pattern: the random access will introduce memory bank conflicts and decrease the throughput (Figure 1b). From the hardware perspective, conventional 3D models are inefficient due to large memory footprint and random memory access.

Hardware-Efficient Deep Learning. Extensive attention has been paid to hardware-efficient deep learning for real-world applications. For instance, researchers have proposed to reduce the memory access cost by pruning and quantizing the models or directly designing the compact models. However, all these approaches are general-purpose and are suitable for arbitrary neural networks. we instead design our efficient primitive based on some domain-specific properties: e.g., 3D point clouds are highly sparse and spatially structured. Voxel-Based 3D Models. Conventionally, researchers relied on the volumetric representation to process the 3D data.

Recent studies suggest that the volumetric representation can also be used in 3D shape segmentation and 3D object detection. Point-Based 3D Models. PointNet takes advantage of the symmetric function to process the unordered point sets in 3D. Later research proposed to stack PointNets hierarchically to model neighborhood information and increase model capacity. Instead of stacking PointNets as basic blocks.



(a) Voxel-based: memory grows cubically

(b) Point-based: large memory/computation overheads

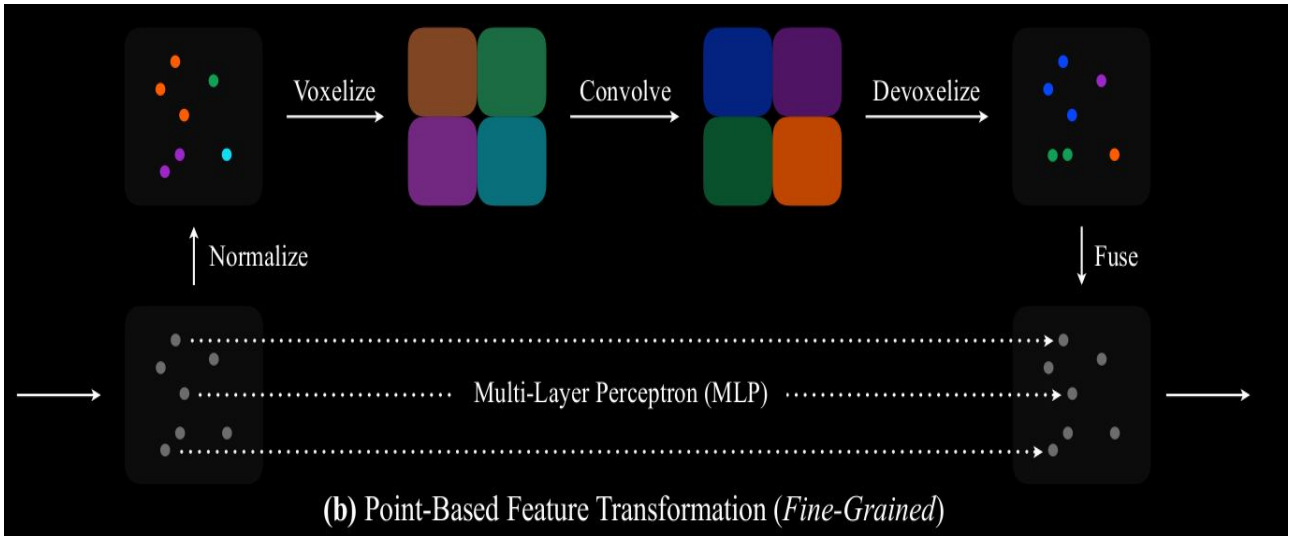
Figure 2: Both voxel-based and point-based NN models are inefficient. Left: the voxel-based model suffers from large information loss at acceptable GPU memory consumption (model: 3D-UNet [51]; dataset: ShapeNet Part [3]). Right: the point-based model suffers from large irregular memory access and dynamic kernel computation overheads.

Voxel-Based Models: Large Memory Footprint:

Voxel-based representation is regular and has good memory locality. However, it requires very high resolution in order not to lose information. When the resolution is low, multiple points are bucketed into the same voxel grid, and these points will no longer be distinguishable. A point is kept only when it exclusively occupies one voxel grid.

Point-Based Models: Irregular Memory Access and Dynamic Kernel Overhead

Point-based 3D modeling methods are memory efficient. The initial attempt, PointNet is also computation efficient, but it lacks the local context modeling capability.



Draft.

Precision (also called positive predictive value) is the fraction of relevant instances among the retrieved instances.

Recall (also known as sensitivity) is the fraction of the total amount of relevant instances that were actually retrieved

12 dogs and some cats. Of the *8 identified as dogs*, *5 actually are dogs* (true positives), while the rest are cats (false positives).

precision is 5/8 while its recall is 5/12.

Point interpolation: to calculate the total area under the horizontal line
So we can calculate AP (Average Precision)

Mean Average Precision: after calculating AP for multiple classes calculate the mean for all the AP(s).

In geometry, the **orientation**اتجاة, **angular position**, **attitude**, or **direction** of an object such as a line, plane or rigid body is part of the description of how it is placed in the space it occupies.[1] More specifically, it refers to the imaginary rotation that is needed to move the object from a reference placement to its current placement.

Average Orientation: is to calculate the new location for the point or object based on average points or directions around it.
Evaluation Metric :
https://github.com/bostondiditeam/kitti/wiki/Evaluation-Metric
Pose Estimation RCNN (For Mask Detection):
https://www.youtube.com/watch?v=nUjGLjOmF7o

Finding all the bodies in frame and then find and identify all the key points (points of interest) of the body then draw lines between those key points so you can sketch those keypoints.

To calculate curves
Precision-Recall Curves:
https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/

Voxel : A **voxel** is a unit of graphic information that defines a point in three-dimensional space.
PointVoxel-CNN (PV-RCNN), for accurate 3D object detection from point clouds. integrates both 3D voxel Convolutional Neural Network (CNN) and PointNet-based set abstraction to learn more discriminative point cloud features. It takes advantage of efficient learning and high-quality proposals of the 3D voxel CNN and the flexible receptive fields of the PointNet-based networks.
summarizes the 3D scene with a 3D voxel CNN into a small set of keypoints via a novel voxel set abstraction module.

3D deep learning is important in multiple fields such as (robotics, VR/AR, Self Driving Cars), So we need to have a good model so it can be efficient from both side time and memory operations.

Note that: Low Resolution leads to significant information loss.
High Resolution leads to large memory consumptions.

Voxel-Based Models : Non-Scalable.
حيزيد جامد بزيادة الـ resolution بتاعة الصورة
Point-Based Models : Spares-Overhead.
دة زى مقارنة بين كل model والتانى يعنى وكل واحد عندة مشاكل بنحاول نختار الأنسب لينا بينهم يعنى
As both last 2 models are not efficient there is a new model we proposed called PV-Conv : Point-Voxel Convolution

Advantages of voxel-based model -> regularity
Advantages of point-based model -> memory footprint
What is Voxelization?

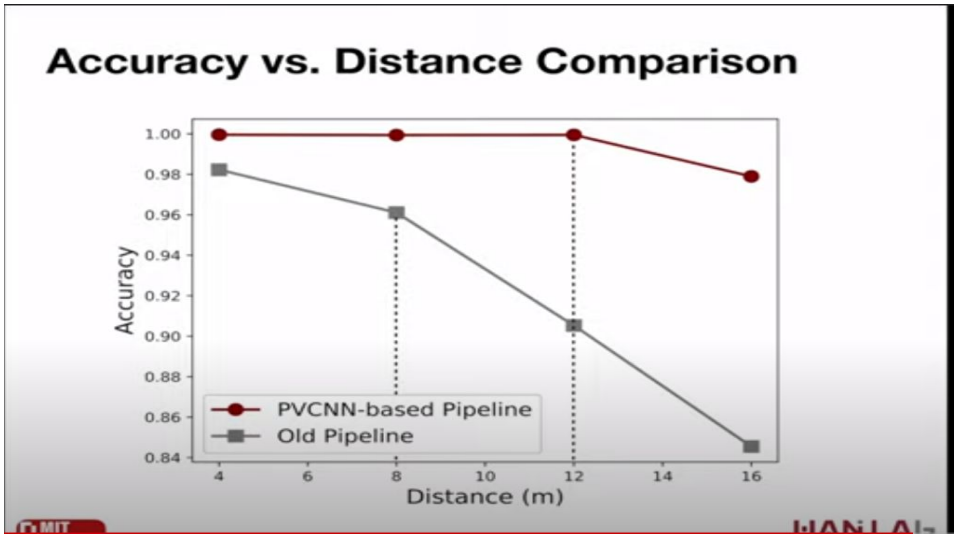
The word Voxel is similar to the word Pixel, where a pixel is in 2D and Voxel stands for a volume in 3D. So, if you take a 3D object and voxelize it, you would end up with a 3D shape of your object. In most cases, a Voxel doesn't know anything about the "shape" within its volume, it only knows if something is there or not (0 or 1).

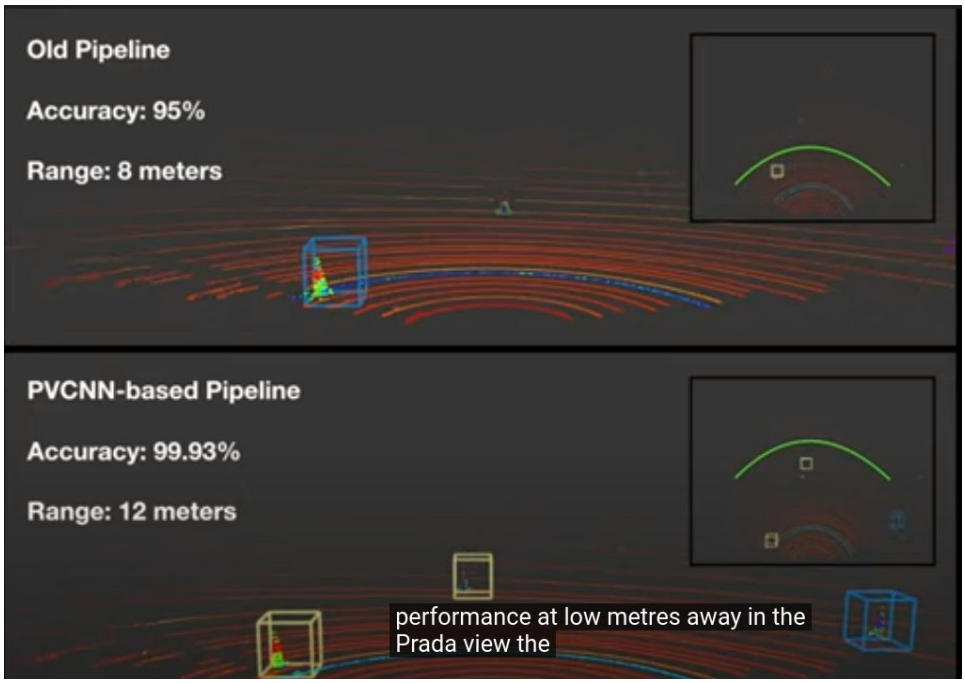
Note that: PVCNN outperforms point CNN with 2.7x measured speed up and 1.5x memory reduction طبعا دة احسن بكتير من اول 2 مودلز

Note that PVCNN is faster than F-Point Net ++ and makes reduction in memory better than it too.

Accuracy for PVCNN is more better than the normal pipeline methods with respect to the distance كل لما المسافة بتزيد كل لما بيكون الطريقة دة احسن

PointVoxel-RCNN (PV-RCNN), for accurate 3D object detection from point clouds.
The method deeply integrates both 3D voxel Convolutional Neural Network (CNN) and PointNet-based set abstraction to learn more discriminative point cloud features





State of art : the most recent stage in the development of a product

مثال على ال object classification باستخدام ال Voxel-Based CNN

https://www.youtube.com/watch?v=a246GAffWZk

بيوضح النسبة بتاعة ال prediction كام فى المية

Code:

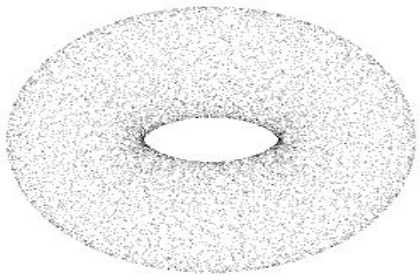
You need python-supported version of Visualization ToolKit(VTK) library. After that, you only need some basic python modules such as PyQt, numpy, etc, and run main.py of the project.

https://github.com/EJShim/EJModelNet

المصطلح دة مهم لانه مستخدم كتير

A point cloud: is a set of data points in space. Point clouds are generally produced by 3D scanners, which measure many points on the external surfaces of objects around them.

Example:



3DmFV: 3D Point Cloud Classification in Real-Time using Convolutional Neural Networks

https://www.youtube.com/watch?v=jLNxXNChwdk

https://github.com/sitzikbs/3DmFV-Net

دة طبعا مش عارف هل حيبقى ليها لزمة ولا اية فحقرر انى مخليش ليها لزمة

Three-Dimensional Point Cloud Classification in Real-Time using Convolutional Neural Networks

https://www.youtube.com/watch?v=KYNDzlcQMWA another funny video XD about mask and pose detection.

If you want to look at something funny at this nightmare

https://www.youtube.com/watch?v=nUjGLjOmF7o

It is a position estimation with tensorflow “Funny Project” you can play a little.

لو عاوز انى اعمل box حوالين ال object فى الصورة زى الكلب او عربية كدة يعنى

سيبك من ان دة مش علينا ولكن وهو بيسطب ال requirements والحاجة ممكن تستفيد منة بعيد

Mask RCNN:

Faster RCNN gives you object detection and localization gives object labels and bounding boxes. “Does this pixel belong to the object or not”.

https://www.youtube.com/watch?v=2TikTv6PWDw

Github slide: https://github.com/markjay4k/Mask-RCN...

Mask RCNN Repo: https://github.com/matterport/Mask_RCNN

Lecture on detection and segmentation “Stanford”: 1H and 15Min Don’t watch it lol :)

https://www.youtube.com/watch?v=nDPWywWRIRo

Maybe it is the tool pytorch.

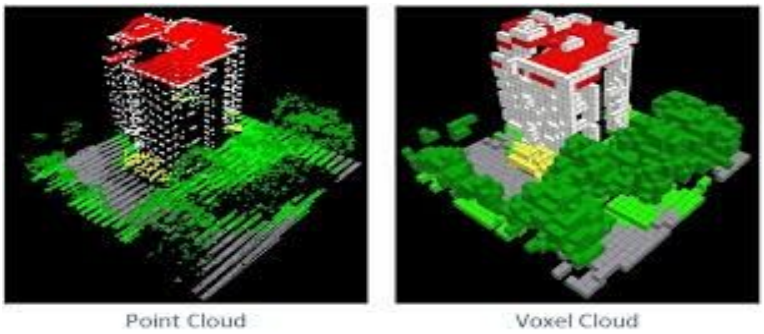
For PointNet-Based object detection

https://www.youtube.com/watch?v=vq0LWzTb-Vo

Project with code 3D Object Detection for Autonomous Driving :

https://github.com/fregu856/3DOD_thesis code

https://www.youtube.com/watch?v=KdrHLXpYYIg video



Point vs voxel cloud

Aggregate View Object Detection (AVOD) network for autonomous driving scenarios. Sequences are sped up for better viewing. For more information and open source code, please see our paper:

https://arxiv.org/abs/1712.02294
https://www.youtube.com/watch?v=mDaqKICiHyA
Code:
https://github.com/kujason/avod
Another try
https://www.youtube.com/watch?v=nE2XtCvPEDk
Paper:
https://arxiv.org/abs/1807.02062
The most beautiful video here :
https://www.youtube.com/watch?v=KXpZ6B1YB_k