Deep Learning for 3D Point Clouds: A Survey

Yulan Guo*, Hanyun Wang*, Qingyong Hu*, Hao Liu*, Li Liu, and Mohammed Bennamoun

Abstract—Point cloud learning has lately attracted increasing attention due to its wide applications in many areas, such as computer vision, autonomous driving, and robotics. As a dominating technique in AI, deep learning has been successfully used to solve various 2D vision problems. However, deep learning on point clouds is still in its infancy due to the unique challenges faced by the processing of point clouds with deep neural networks. Recently, deep learning on point clouds has become even thriving, with numerous methods being proposed to address different problems in this area. To stimulate future research, this paper presents a comprehensive review of recent progress in deep learning methods for point clouds. It covers three major tasks, including 3D shape classification, 3D object detection and tracking, and 3D point cloud segmentation. It also presents comparative results on several publicly available datasets, together with insightful observations and inspiring future research directions.

Index Terms—deep learning, point clouds, 3D data, shape classification, shape retrieval, object detection, object tracking, scene flow, instance segmentation, semantic segmentation, part segmentation.

1 Introduction

With the rapid development of 3D acquisition technologies, 3D sensors are becoming increasingly available and affordable, including various types of 3D scanners, LiDARs, and RGB-D cameras (such as Kinect, RealSense and Apple depth cameras) [1]. 3D data acquired by these sensors can provide rich geometric, shape and scale information [2], [3]. Complemented with 2D images, 3D data provides an opportunity for a better understanding of the surrounding environment for machines. 3D data has numerous applications in different areas, including autonomous driving, robotics, remote sensing, and medical treatment [4].

3D data can usually be represented with different formats, including depth images, point clouds, meshes, and volumetric grids. As a commonly used format, point cloud representation preserves the original geometric information in 3D space without any discretization. Therefore, it is the preferred representation for many scene understanding related applications such as autonomous driving and robotics. Recently, deep learning techniques have dominated many research areas, such as computer vision, speech recognition, and natural language processing. However, deep learning on 3D point clouds still face several significant challenges [5], such as the small scale of datasets, the high dimensionality and the unstructured nature of 3D point clouds. On this basis, this paper focuses on the analysis of deep learning methods which have been used to process 3D point clouds.

Deep learning on point clouds has been attracting more and more attention, especially in the last five years. Several publicly available datasets are also released, such as ModelNet [6], ScanObjectNN [7], ShapeNet [8], PartNet [9], S3DIS [10], ScanNet [11], Semantic3D [12], ApolloCar3D [13], and the KITTI Vision Benchmark Suite [14], [15]. These datasets have further boosted the research of deep learning on 3D point clouds, with an increasingly number of methods being proposed to address various problems related to point cloud processing, including 3D shape classification, 3D object detection and tracking, 3D point cloud segmentation, 3D point cloud registration, 6-DOF pose estimation, and 3D reconstruction [16], [17], [18]. Few surveys of deep learning on 3D data are also available, such as [19], [20], [21], [22]. However, our paper is the first to specifically focus on deep learning methods for point cloud understanding. A taxonomy of existing deep learning methods for 3D point clouds is shown in Fig. 1.

深度学习用于3D点云面临的问题：1 数据少　2数据高维度　3 非结构化

3D点云数据集

Compared with the existing literatures, the major contributions of this work can be summarized as follows:

1) To the best of our knowledge, this is the *first* survey paper to comprehensively cover deep learning methods for several important point cloud understanding tasks, including 3D shape classification, 3D object detection and tracking, and 3D point cloud segmentation.
2) As opposed to existing reviews [19], [20], we specifically focus on deep learning methods for *3D point clouds* rather than all types of 3D data.
3) This paper covers the *most recent and advanced progresses* of deep learning on point clouds. Therefore, it provides the readers with the state-of-the-art methods.
4) Comprehensive *comparisons of existing methods* on several publicly available datasets are provided (e.g., in Tables 2, 3, 4, 5), with brief summaries and insightful discussions being presented.

The structure of this paper is as follows. Section 2 introduces the datasets and evaluation metrics for the respective tasks. Section 3 reviews the methods for 3D shape classification. Section 4 provides a survey of existing methods for 3D object detection and tracking. Section 5

- *Y Guo and H. Liu are with the School of Electronics and Communication Engineering, Sun Yat-sen University, China. H. Wang is with the School of Surveying and Mapping, Information Engineering University, China. Q. Hu is with Department of Computer Science, University of Oxford, UK. L. Liu is with the College of System Engineering, National University of Defense Technology, China, and also with the Center for Machine Vision and Signal Analysis, University of Oulu, Finland. M. Bennamoun is with the Department of Computer Science and Software Engineering, the University of Western Australia, Australia.*
- **Y. Guo, H. Wang, Q. Hu and H. Liu have equal contribution to this work and are co-first authors.*
- *Corresponding author: Yulan Guo (yulan.guo@nudt.edu.cn).*

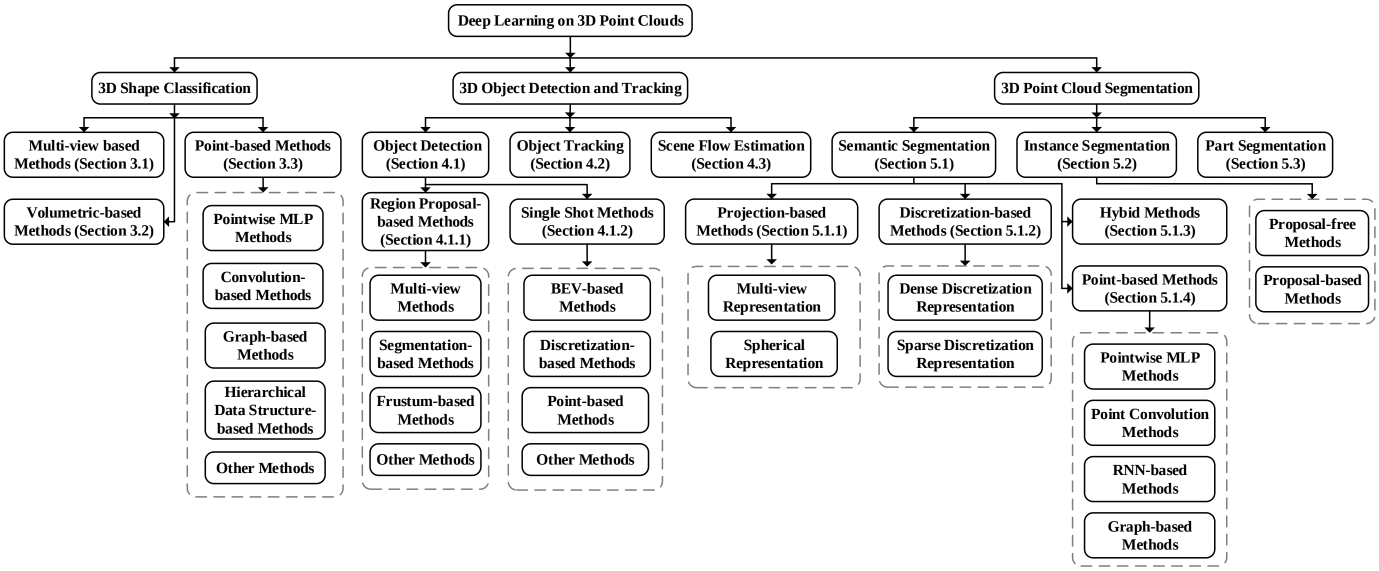

Deep Learning on 3D Point Clouds
3D Shape Classification
3D Object Detection and Tracking
3D Point Cloud Segmentation
Multi-view based Methods (Section 3.1)
Volumetric-based Methods (Section 3.2)
Point-based Methods (Section 3.3)
Pointwise MLP Methods
Convolution-based Methods
Graph-based Methods
Hierarchical Data Structure-based Methods
Other Methods
Object Detection (Section 4.1)
Region Proposal-based Methods (Section 4.1.1)
Multi-view Methods
Segmentation-based Methods
Frustum-based Methods
Other Methods
Single Shot Methods (Section 4.1.2)
BEV-based Methods
Discretization-based Methods
Point-based Methods
Other Methods
Object Tracking (Section 4.2)
Scene Flow Estimation (Section 4.3)
Semantic Segmentation (Section 5.1)
Projection-based Methods (Section 5.1.1)
Multi-view Representation
Spherical Representation
Discretization-based Methods (Section 5.1.2)
Dense Discretization Representation
Sparse Discretization Representation
Hybid Methods (Section 5.1.3)
Point-based Methods (Section 5.1.4)
Pointwise MLP Methods
Point Convolution Methods
RNN-based Methods
Graph-based Methods
Instance Segmentation (Section 5.2)
Part Segmentation (Section 5.3)
Proposal-free Methods
Proposal-based Methods

Fig. 1: A taxonomy of deep learning methods for 3D point clouds.

presents a review of methods for point cloud segmentation, including semantic segmentation, instance segmentation, and part segmentation. Finally, Section 6 concludes the paper. We also provide a regularly updated project page on: https://github.com/QingyongHu/SoTA-Point-Cloud.

2 Background

2.1 Datasets

A large number of datasets have been collected to evaluate the performance of deep learning algorithms for different 3D point clouds applications. Table 1 lists some typical datasets used for 3D shape classification, 3D object detection and tracking, and 3D point cloud segmentation. In particular, the attributes of these datasets are also summarized.

For 3D shape classification, there are two types of datasets: synthetic datasets [6], [8] and real-world datasets [7], [11]. Objects in the synthetic datasets are complete, without any occlusion and background. In contrast, objects in the real-world datasets are occluded at different levels and some objects are contaminated with background noise.

For 3D object detection and tracking, there are two types of datasets: indoor scenes [11], [25] and outdoor urban scenes [14], [28], [30], [31]. The point clouds in the indoor datasets are either converted from dense depth maps or sampled from 3D meshes. The outdoor urban datasets are designed for autonomous driving, where objects are spatially well separated and these point clouds are sparse.

For 3D point cloud segmentation, these datasets are acquired by different types of sensors, including Mobile Laser Scanners (MLS) [15], [34], [36], Aerial Laser Scanners (ALS) [33], [38], static Terrestrial Laser Scanners (TLS) [12], RGB-D cameras [11] and other 3D scanners [10]. These datasets can be used to develop algorithms for various challenges including similar distractors, shape incompleteness, and class imbalance.

2.2 Evaluation Metrics

Different evaluation metrics have been proposed to test these methods for various point cloud understanding tasks. For 3D shape classification, *Overall Accuracy* (OA) and *mean class accuracy* (mAcc) are the most frequently used performance criteria. 'OA' represents the mean accuracy for all test instances and 'mAcc' represents the mean accuracy for all shape classes. For 3D object detection, *Average Precision* (AP) is the most frequently used criterion. It is calculated as the area under the precision-recall curve. *Precision* and *Success* are commonly used to evaluate the overall performance of a 3D single object tracker. *Average Multi-Object Tracking Accuracy* (AMOTA) and *Average Multi-Object Tracking Precision* (AMOTP) are the most frequently used criteria for the evaluation of 3D multi-object tracking. For 3D point cloud segmentation, OA, *mean Intersection over Union* (mIoU) and *mean class Accuracy* (mAcc) [10], [12], [15], [36], [37] are the most frequently used criteria for performance evaluation. In particular, *mean Average Precision* (mAP) [39] is also used in instance segmentation of 3D point clouds.

3 3D Shape Classification

Methods for this task usually learn the embedding of each point first and then extract a global shape embedding from the whole point cloud using an aggregation method. Classification is finally achieved by feeding the global embedding into several fully connected layers. According to the data type of input for neural networks, existing 3D shape classification methods can be divided into multi-view based, volumetric-based and point-based methods. Several milestone methods are illustrated in Fig. 2.

Multi-view based methods project an unstructured point cloud into 2D images, while volumetric-based methods convert a point cloud into a 3D volumetric representation. Then, well-established 2D or 3D convolutional networks are leveraged to achieve shape classification. In contrast, point-based methods directly work on raw point clouds without any voxelization or projection. Point-based methods do not

TABLE 1: A summary of existing datasets for 3D shape classification, 3D object detection and tracking, and 3D point cloud segmentation. [1] The number of classes used for evaluation and the number of annotated classes (shown in brackets).

Datasets for 3D Shape Classification							
Name and Reference	Year	#Samples	#Classes	#Training	#Test	Type	Representation
McGill Benchmark [23]	2008	456	19	304	152	Synthetic	Mesh
Sydney Urban Objects [24]	2013	588	14	-	-	Real-World	Point Clouds
ModelNet10 [6]	2015	4899	10	3991	605	Synthetic	Mesh
ModelNet40 [6]	2015	12311	40	9843	2468	Synthetic	Mesh
ShapeNet [8]	2015	51190	55	-	-	Synthetic	Mesh
ScanNet [11]	2017	12283	17	9677	2606	Real-World	RGB-D
ScanObjectNN [7]	2019	2902	15	2321	581	Real-World	Point Clouds
Datasets for 3D Object Detection and Tracking							
Name and Reference	Year	#Scenes	#Classes	#Annotated Frames	#3D Boxes	Secne Type	Sensors
KITTI [14]	2012	22	8	15K	200K	Urban (Driving)	RGB & LiDAR
SUN RGB-D [25]	2015	47	37	5K	65K	Indoor	RGB-D
ScanNetV2 [11]	2018	1.5K	18	-	-	Indoor	RGB-D & Mesh
H3D [26]	2019	160	8	27K	1.1M	Urban (Driving)	RGB & LiDAR
Argoverse [27]	2019	113	15	44K	993K	Urban (Driving)	RGB & LiDAR
Lyft L5 [28]	2019	366	9	46K	1.3M	Urban (Driving)	RGB & LiDAR
A*3D [29]	2019	-	7	39K	230K	Urban (Driving)	RGB & LiDAR
Waymo Open [30]	2020	1K	4	200K	12M	Urban (Driving)	RGB & LiDAR
nuScenes [31]	2020	1K	23	40K	1.4M	Urban (Driving)	RGB & LiDAR
Datasets for 3D Point Cloud Segmentation							
Name and Reference	Year	#Points	#Classes[1]	#Scans	Spatial Size	RGB	Sensors
Oakland [32]	2009	1.6M	5(44)	17	-	N/A	MLS
ISPRS [33]	2012	1.2M	9	-	-	N/A	ALS
Paris-rue-Madame [34]	2014	20M	17	2	-	N/A	MLS
IQmulus [35]	2015	300M	8(22)	10	-	N/A	MLS
ScanNet [11]	2017	-	20(20)	1513	8×4×4	Yes	RGB-D
S3DIS [10]	2017	273M	13(13)	272	10×5×5	Yes	Matterport
Semantic3D [12]	2017	4000M	8(9)	15/15	250×260×80	Yes	TLS
Paris-Lille-3D [36]	2018	143M	9(50)	3	200×280× 30	N/A	MLS
SemanticKITTI [15]	2019	4549M	25(28)	23201/20351	150×100×10	N/A	MLS
Toronto-3D [37]	2020	78.3M	8(9)	4	260×350× 40	Yes	MLS
DALES [38]	2020	505M	8(9)	40	500×500×65	N/A	ALS

introduce explicit information loss and become increasingly popular. Note that, this paper mainly focuses on point-based methods, but also includes few multi-view based and volumetric-based methods for completeness.

3.1 Multi-view based Methods

These methods first project a 3D shape into multiple views and extract view-wise features, and then fuse these features for accurate shape classification. How to aggregate multiple view-wise features into a discriminative global representation is a key challenge for these methods.

MVCNN [40] is a pioneering work, which simply max-pools multi-view features into a global descriptor. However, max-pooling only retains the maximum elements from a specific view, resulting in information loss. MHBN [41] integrates local convolutional features by harmonized bilinear pooling to produce a compact global descriptor. Yang et al. [42] first leveraged a relation network to exploit the inter-relationships (e.g., region-region relationship and view-view relationship) over a group of views, and then aggregated these views to obtain a discriminative 3D object representation. In addition, several other methods [43], [44], [45], [46] have also been proposed to improve the recognition accuracy. Unlike previous methods, Wei et al. [47] used a directed graph in View-GCN by considering multiple views as grpah nodes. The core layer composing of local graph convolution, non-local message passing and selective view-sampling is then applied to the constructed graph. The concatenation of max-pooled node features at all levels is finally used to form the global shape descriptor.

3.2 Volumetric-based Methods

These methods usually voxelize a point cloud into 3D grids, and then apply a 3D Convolution Neural Network (CNN) on the volumetric representation for shape classification.

Maturana et al. [48] introduced a volumetric occupancy network called VoxNet to achieve robust 3D object recognition. Wu et al. [6] proposed a convolutional deep belief-based 3D ShapeNets to learn the distribution of points from various 3D shapes (which are represented by a probability distribution of binary variables on voxel grids). Although encouraging performance has been achieved, these methods are unable to scale well to dense 3D data since the computation and memory footprint grow cubically with the resolution.

To this end, a hierarchical and compact structure (such as octree) is introduced to reduce the computational and memory costs of these methods. OctNet [49] first hierarchically partitions a point cloud using a hybrid grid-octree structure, which represents the scene with several shallow octrees along a regular grid. The structure of octree is encoded efficiently using a bit string representation, and the feature vector of each voxel is indexed by simple arithmetic. Wang et al. [50] proposed an Octree-based CNN for 3D shape classification. The average normal vectors of a 3D model sampled in the finest leaf octants are fed into the network, and 3D-CNN is applied on the octants occupied

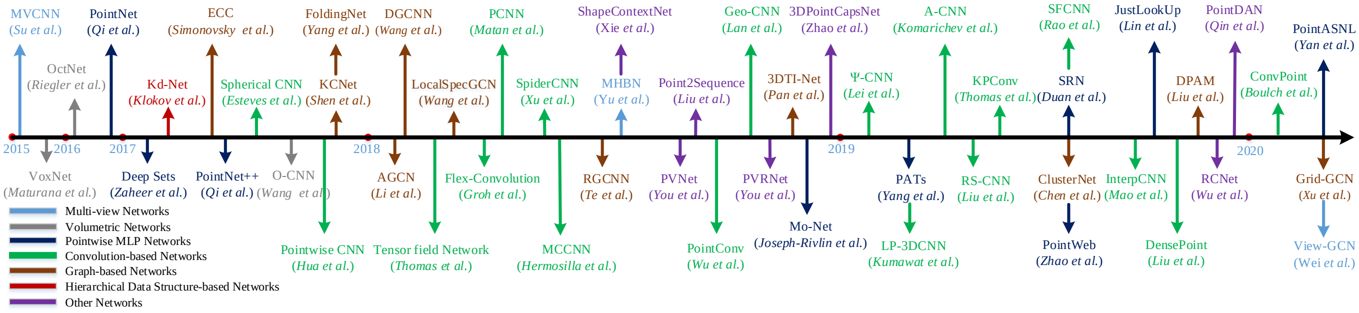


Fig. 2: Chronological overview of the most relevant deep learning-based 3D shape classification methods.

by the 3D shape surface. Compared to a baseline network based on dense input grids, OctNet requires much less memory and runtime for high-resolution point clouds. Le et al. [51] proposed a hybrid network called PointGrid, which integrates the point and grid representation for efficient point cloud processing. A constant number of points is sampled within each embedding volumetric grid cell, which allows the network to extract geometric details by using 3D convolutions. Ben-Shabat et al. [52] transformed the input point cloud into 3D grids which are further represented by 3D modified Fisher Vector (3DmFV) method, and then learned the global representation through a conventional CNN architecture.

3.3 Point-based Methods

According to the network architecture used for the feature learning of each point, methods in this category can be divided into pointwise MLP, convolution-based, graph-based, hierarchical data structure-based methods and other typical methods.

3.3.1 Pointwise MLP Methods

These methods model each point independently with several shared Multi-Layer Perceptrons (MLPs) and then aggregate a global feature using a symmetric aggregation function, as shown in Fig. 3.

Typical deep learning methods for 2D images cannot be directly applied to 3D point clouds due to their inherent data irregularities. As a pioneering work, PointNet [5] directly takes point clouds as its input and achieves permutation invariance with a symmetric function. Specifically, PointNet learns pointwise features independently with several MLP layers and extracts global features with a max-pooling layer. Deep sets [53] achieves permutation invariance by summing up all representations and applying nonlinear transformations. Since features are learned independently for each point in PointNet [5], the local structural information between points cannot be captured. Therefore, Qi et al. [54] proposed a hierarchical network PointNet++ to capture fine geometric structures from the neighborhood of each point. As the core of PointNet++ hierarchy, its set abstraction level is composed of three layers: the sampling layer, the grouping layer and the PointNet based learning layer. By stacking several set abstraction levels, PointNet++ learns features from a local geometric structure and abstracts the local features layer by layer.

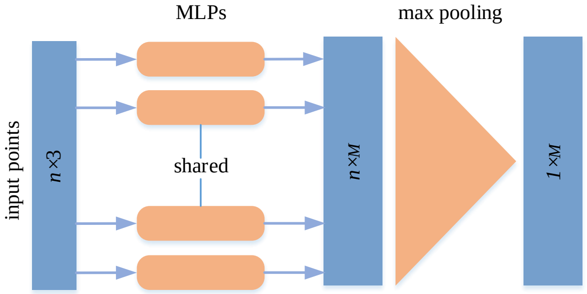


Fig. 3: A lightweight architecture of PointNet. n denotes the number of input points, M denotes the dimension of the learned features for each point.

Because of its simplicity and strong representation ability, many networks have been developed based on PointNet [5]. The architecture of Mo-Net [55] is similar to PointNet [5] but it takes a finite set of moments as its input. Point Attention Transformers (PATs) [56] represents each point by its own absolute position and relative positions with respect to its neighbors and learns high dimensional features through MLPs. Then, Group Shuffle Attention (GSA) is used to capture relations between points, and a permutation invariant, differentiable and trainable end-to-end Gumbel Subset Sampling (GSS) layer is developed to learn hierarchical features. Based on PointNet++ [54], PointWeb [57] utilizes the context of the local neighborhood to improve point features using Adaptive Feature Adjustment (AFA). Duan et al. [58] proposed a Structural Relational Network (SRN) to learn structural relational features between different local structures using MLP. Lin et al. [59] accelerated the inference process by constructing a lookup table for both input and function spaces learned by PointNet. The inference time on the ModelNet and ShapeNet datasets is sped up by 1.5 ms and 32 times over PointNet on a moderate machine. SRINet [60] first projects a point cloud to obtain rotation invariant representations, and then utilizes PointNet-based backbone to extract a global feature and graph-based aggregation to extract local features. In PointASNL, Yan et al. [61] utilized an Adaptive Sampling (AS) module to adaptively adjust the coordinates and features of points sampled by the Furthest Point Sampling (FPS) algorithm, and proposed a local-nonlocal (L-NL) module to capture the local and long range dependencies of these sampled points.

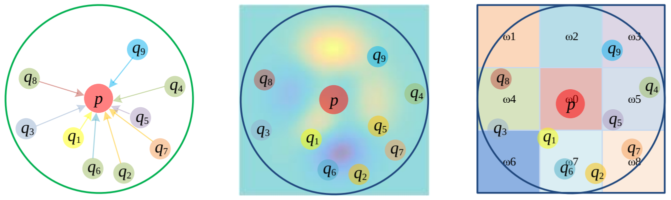
3.3.2 Convolution-based Methods

Compared with kernels defined on 2D grid structures (e.g., images), convolutional kernels for 3D point clouds are hard to design due to the irregularity of point clouds. According to the type of convolutional kernels, current 3D convolution methods can be divided into continuous and discrete convolution methods, as shown in Fig. 4.

3D Continuous Convolution Methods. These methods define convolutional kernels on a continuous space, where the weights for neighboring points are related to the spatial distribution with respect to the center point.

3D convolution can be interpreted as a weighted sum over a given subset. As the core layer of RS-CNN [62], RS-Conv takes a local subset of points around a certain point as its input, and the convolution is implemented using an MLP by learning the mapping from low-level relations (such as Euclidean distance and relative position) to high-level relations between points in the local subset. In [63], kernel elements are selected randomly in a unit sphere. An MLP-based continuous function is then used to establish relation between the locations of the kernel elements and the point cloud. In DensePoint [64], convolution is defined as a Single-Layer Perceptron (SLP) with a nonlinear activator. Features are learned by concatenating features from all previous layers to sufficiently exploit the contextual information. Thomas et al. [65] proposed both rigid and deformable Kernel Point Convolution (KPConv) operators for 3D point clouds using a set of learnable kernel points. ConvPoint [66] separates the convolution kernel into spatial and feature parts. The locations of the spatial part are randomly selected from a unit sphere and the weighting function is learned through a simple MLP.

Some methods also use existing algorithms to perform convolution. In PointConv [67], convolution is defined as a Monte Carlo estimation of the continuous 3D convolution with respect to an importance sampling. The convolutional kernels consist of a weighting function (which is learned with MLP layers) and a density function (which is learned by a kernelized density estimation and an MLP layer). To improve memory and computational efficiency, the 3D convolution is further reduced into two operations: matrix multiplication and 2D convolution. With the same parameter setting, its memory consumption can be reduced by about 64 times. In MCCNN [68], convolution is considered as a Monte Carlo estimation process relying on a sample's density function (which is implemented with MLP). Poisson disk sampling is then used to construct a point cloud hierarchy. This convolution operator can be used to perform convolution between two or multiple sampling methods and can handle varying sampling densities. In SpiderCNN [69], SpiderConv is proposed to define convolution as the product of a step function and a Taylor expansion defined on the k nearest neighbors. The step function captures the coarse geometry by encoding the local geodesic distance, and the Taylor expansion captures the intrinsic local geometric variations by interpolating arbitrary values at the vertices of a cube. Besides, a convolution network PCNN [70] is also proposed for 3D point clouds based on the radial basis function.



(a) 3D neighboring points (b) 3D continuous convolution (c) 3D discrete convolution

Fig. 4: An illustration of a continuous and discrete convolution for local neighbors of a point. (a) represents a local neighborhood q_i centered at point p; (b) and (c) represent 3D continuous and discrete convolution, respectively.

Several methods have been proposed to address the rotation equivariant problem faced by 3D convolution networks. Esteves et al. [71] proposed 3D Spherical CNN to learn rotation equivariant representation for 3D shapes, which takes multi-valued spherical functions as its input. Localized convolutional filters are obtained by parameterizing spectrum with anchor points in the spherical harmonic domain. Tensor field networks [72] are proposed to define the point convolution operation as the product of a learnable radial function and spherical harmonics, which are locally equivariant to 3D rotations, translations, and permutations. The convolution in [73] is defined based on the spherical cross-correlation and implemented using a generalized Fast Fourier Transformation (FFT) algorithm. Based on PCNN, SPHNet [74] achieves rotation invariance by incorporating spherical harmonic kernels during convolution on volumetric functions.

To accelerate computing speed, Flex-Convolution [75] defines weights of convolution kernel as standard scalar product over k nearest neighbors, which can be accelerated using CUDA. Experimental results have demonstrated its competitive performance on a small dataset with fewer parameters and lower memory consumption.

3D Discrete Convolution Methods. These methods define convolutional kernels on regular grids, where the weights for neighboring points are related to the offsets with respect to the center point.

Hua et al. [76] transformed non-uniform 3D point clouds into uniform grids and defined convolutional kernels on each grid. The proposed 3D kernel assigns the same weights to all points falling into the same grid. For a given point, the mean features of all the neighboring points that are located on the same grid are computed from the previous layer. Then, mean features of all grids are weighted and summed to produce the output of the current layer. Lei et al. [77] defined a spherical convolutional kernel by partitioning a 3D spherical neighboring region into multiple volumetric bins and associating each bin with a learnable weighting matrix. The output of the spherical convolutional kernel for a point is determined by the non-linear activation of the mean of weighted activation values of its neighboring points. In GeoConv [78], the geometric relationship between a point and its neighboring points is explicitly modeled based on six bases. Edge features along each direction of the basis are weighted independently by a direction-associated learnable matrix. These direction-associated features are then aggregated according to the angles formed by the given point and its neighboring points. For a given point, its feature at the current layer is defined as the sum of features of the given point and its neighboring edge features at the previous layer.

PointCNN [79] transforms the input points into a latent and potentially canonical order through a χ-conv transformation (which is implemented through MLP) and then applies typical convolutional operator on the transformed features. By interpolating point features to neighboring discrete convolutional kernel-weight coordinates, Mao et al. [80] proposed an interpolated convolution operator InterpConv to measure the geometric relations between input point clouds and kernel-weight coordinates. Zhang et al. [81] proposed a RIConv operator to achieve rotation invariance, which takes low-level rotation invariant geometric features as input and then turns the convolution into 1D by a simple binning approach. A-CNN [82] defines an annular convolution by looping the array of neighbors with respect to the size of kernel on each ring of the query point and learns the relationship between neighboring points in a local subset.

To reduce the computational and memory cost of 3D CNNs, Kumawat et al. [83] proposed a Rectified Local Phase Volume (ReLPV) block to extract phase in a 3D local neighborhood based on 3D Short Term Fourier Transform (STFT), which significantly reduces the number of parameters. In SFCNN [84], a point cloud is projected onto regular icosahedral lattices with aligned spherical coordinates. Convolutions are then conducted upon the features concatenated from vertices of spherical lattices and their neighbors through convolution-maxpooling-convolution structures. SFCNN is resistant to rotations and perturbations.

3.3.3 Graph-based Methods

Graph-based networks consider each point in a point cloud as a vertex of a graph, and generate directed edges for the graph based on the neighbors of each point. Feature learning is then performed in spatial or spectral domains [85]. A typical graph-based network is shown in Fig. 5.

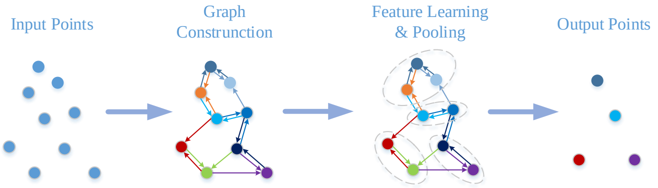


Fig. 5: An illustration of a graph-based network.

Graph-based Methods in Spatial Domain. These methods define operations (e.g., convolution and pooling) in spatial domain. Specifically, convolution is usually implemented through MLP over spatial neighbors, and pooling is adopted to produce a new coarsened graph by aggregating information from each point's neighbors. Features at each vertex are usually assigned with coordinates, laser intensities or colors, while features at each edge are usually assigned with geometric attributes between two connected points.

As a pioneering work, Simonovsky et al. [85] considered each point as a vertex of the graph, and connected each vertex to all its neighbors by a directed edge. Then, Edge-Conditioned Convolution (ECC) is proposed using a filter-generating network (e.g., MLP). Max pooling is adopted to aggregate neighborhood information and graph coarsening is implemented based on VoxelGrid [86]. In DGCNN [87], a graph is constructed in the feature space and dynamically updated after each layer of the network. As the core layer of EdgeConv, an MLP is used as the feature learning function for each edge, and channel-wise symmetric aggregation is applied onto the edge features associated with the neighbors of each point. Further, LDGCNN [88] removes the transformation network and links the hierarchical features from different layers in DGCNN [87] to improve its performance and reduce the model size. An end-to-end unsupervised deep AutoEncoder network (namely, FoldingNet [89]) is also proposed to use the concatenation of a vectorized local covariance matrix and point coordinates as its input. Inspired by Inception [90] and DGCNN [87], Hassani and Haley [91] proposed an unsupervised multi-task autoencoder to learn point and shape features. The encoder is constructed based on mutli-scale graphs. The decoder is constructed using three unsupervised tasks including clustering, self-supervised classification and reconstruction, which are trained jointly with a mutli-task loss. Liu et al. [92] proposed a Dynamic Points Agglomeration Module (DPAM) based on graph convolution to simplify the process of points agglomeration (sampling, grouping and pooling) into a simple step, which is implemented through multiplication of the agglomeration matrix and points feature matrix. Based on the PointNet architecture, a hierarchical learning architecture is constructed by stacking multiple DPAMs. Compared with the hierarchy strategy of PointNet++ [54], DPAM dynamically exploits the relation of points and agglomerates points in a semantic space.

To exploit the local geometric structures, KCNet [93] learns features based on kernel correlation. Specifically, a set of learnable points characterizing geometric types of local structures are defined as kernels. Then, affinity between the kernel and the neighborhood of a given point is calculated. In G3D [94], convolution is defined as a variant of polynomial of adjacency matrix, and pooling is defined as multiplying the Laplacian matrix and the vertex matrix by a coarsening matrix. ClusterNet [95] utilizes a rigorously rotation-invariant module to extract rotation-invariant features from k nearest neighbors for each point, and constructs hierarchical structures of a point cloud based on the unsupervised agglomerative hierarchical clustering method with ward-linkage criteria [96]. The features in each sub-cluster are first learned through an EdgeConv block and then aggregated through max pooling.

To address the time-consuming problem of current data structuring methods (such as FPS and neighbor points querying), Xu et al. [97] proposed to blend the advantages of volumetric based and point based methods to improve the computational efficiency. Experiments on the ModelNet classification task demonstrate that the computational efficiency of the proposed Grid-GCN network is 5× faster than other models in average.

Graph-based Methods in Spectral Domain. These methods define convolutions as spectral filtering, which is implemented as the multiplication of signals on graph with eigenvectors of the graph Laplacian matrix [98], [99].

RGCNN [100] constructs a graph by connecting each point with all other points in the point cloud and updates the graph Laplacian matrix in each layer. To make features of adjacent vertices more similar, a graph-signal smoothness prior is added into the loss function. To address the challenges caused by diverse graph topology of data, the SGC-LL layer in AGCN [101] utilizes a learnable distance metric to parameterize the similarity between two vertices on the graph. The adjacency matrix obtained from graph is normalized using Gaussian kernels and learned distances. HGNN [102] builds a hyperedge convolutional layer by applying spectral convolution on a hypergraph.

Aforementioned methods operate on full graphs. To exploit local structural information, Wang et al. [103] proposed an end-to-end spectral convolution network LocalSpecGCN to work on a local graph (which is constructed from the k nearest neighbors). This method does not require any offline computation of the graph Laplacian matrix and graph coarsening hierarchy. In PointGCN [104], a graph is constructed based on k nearest neighbors from a point cloud and each edge is weighted using a Gaussian kernel. Convolutional filters are defined as Chebyshev polynomials in graph spectral domain. Global pooling and multi-resolution pooling are used to capture global and local features of the point cloud. Pan et al. [105] proposed 3DTI-Net by applying convolution on the k nearest neighboring graphs in spectral domain. The invariance to geometry transformation is achieved by learning from relative Euclidean and direction distances.

3.3.4 Hierarchical Data Structure-based Methods

These networks are constructed based on different hierarchical data structures (e.g., octree and kd-tree). In these methods, point features are learned hierarchically from leaf nodes to the root node along a tree.

Lei et al. [77] proposed an octree guided CNN using spherical convolutional kernels (as described in Section 3.3.2). Each layer of the network corresponds to one layer of the octree and a spherical convolutional kernel is applied at each layer. The values of neurons in the current layer are determined as the mean values of all relevant children nodes in the previous layer. Unlike OctNet [49] which is based on octree, Kd-Net [106] is built using multiple K-d trees with different splitting directions at each iteration. Following a bottom-up approach, the representation of a non-leaf node is computed from representations of its children using MLP. The feature of the root node (which describes the whole point cloud) is finally fed to fully connected layers to predict classification scores. Note that, Kd-Net shares parameters at each level according to the splitting type of nodes. 3DContextNet [107] uses a standard balanced K-d tree to achieve feature learning and aggregation. At each level, point features are first learned through MLP based on local cues (which models inter-dependencies between points in a local region) and global contextual cues (which models the relationship for one position with respect to all other positions). Then, the feature of a non-leaf node is computed from its child nodes using MLP and aggregated by max pooling. For classification, the above process is repeated until the root node is attained.

The hierarchy of SO-Net network is constructed by performing point-to-node k nearest neighbor search [108]. Specifically, a modified permutation invariant Self-Organizing Map (SOM) is used to model the spatial distribution of a point cloud. Individual point features are learned from normalized point-to-node coordinates through a series of fully connected layers. The feature of each node in SOM is extracted from point features associated with this node using channel-wise max pooling. The final feature is then learned from node features using an approach similar to PointNet [5]. Compared to PointNet++ [54], the hierarchy of SOM is more efficient and the spatial distribution of the point cloud is fully explored.

3.3.5 Other Methods

In addition, many other schemes have also been proposed. RBFNet [113] explicitly models the spatial distribution of points by aggregating features from sparsely distributed Radial Basis Function (RBF) kernels with learnable kernel positions and sizes. 3DPointCapsNet [112] learns point independent features with pointwise MLP and convolutional layers, and extracts global latent representation with multiple max-pooling layers. Based on unsupervised dynamic routing, powerful representative latent capsules are then learned. Qin et al. [116] proposed an end-to-end unsupervised domain adaptation network PointDAN for 3D point cloud representation. To capture semantic properties of a point cloud, a self-supervised method is proposed to reconstruct the point cloud, whose parts have been randomly rearranged [117]. Li et al. [118] proposed an auto-augmentation framework, PointAugment, to automatically optimize and augment point cloud samples for network training. Specifically, shape-wise transformation and point-wise displacement for each input sample are automatically learned, and the network is trained by alternatively optimizing and updating the learnable parameters of its augmentor and classifier. Inspired by shape context [119], Xie et al. [109] proposed a ShapeContextNet architecture by combining affinity point selection and compact feature aggregation into a soft alignment operation using dot-product self-attention [120]. To handle noise and occlusion in 3D point clouds, Bobkov et al. [121] fed handcrafted point pair function based 4D rotation invariant descriptors into a 4D convolutional neural network. Prokudin et al. [122] first randomly sampled a basis point set with a uniform distribution from a unit ball, and then encoded a point cloud as minimal distances to the basis point set. Consequently, the point cloud is converted to a vector with a relatively small fixed length. The encoded representation can then be processed with existing machine learning methods.

RCNet [115] utilizes standard RNN and 2D CNN to construct a permutation-invariant network for 3D point cloud processing. The point cloud is first partitioned into parallel beams and sorted along a specific dimension, and each beam is then fed into a shared RNN. The learned features are further fed into an efficient 2D CNN for hierarchical feature aggregation. To enhance its description ability, RCNet-E is proposed to ensemble multiple RCNets along different partition and sorting directions. Point2Sequences [114] is another RNN-based model that captures correlations between different areas in local regions of point clouds. It considers features learned from a local region at multiple scales as sequences and feeds these sequences from all local regions into an RNN-based encoder-decoder structure to aggregate local region features.

TABLE 2: Comparative 3D shape classification results on the ModelNet10/40 benchmarks. Here, we only focus on point-based networks. '#params' represents the number of parameters of a model, 'OA' represents the mean accuracy for all test instances and 'mAcc' represents the mean accuracy for all shape classes in the table. The symbol '-' means the results are unavailable.

Methods		Input	#params (M)	ModelNet40 (OA)	ModelNet40 (mAcc)	ModelNet10 (OA)	ModelNet10 (mAcc)
Pointwise MLP Methods	PointNet [5]	Coordinates	3.48	89.2%	86.2%	-	-
	PointNet++ [54]	Coordinates	1.48	90.7%	-	-	-
	MO-Net [55]	Coordinates	3.1	89.3%	86.1%	-	-
	Deep Sets [53]	Coordinates	-	87.1%	-	-	-
	PAT [56]	Coordinates	-	91.7%	-	-	-
	PointWeb [57]	Coordinates	-	92.3%	89.4%	-	-
	SRN-PointNet++ [58]	Coordinates	-	91.5%	-	-	-
	JUSTLOOKUP [59]	Coordinates	-	89.5%	86.4%	92.9%	92.1%
	PointASNL [61]	Coordinates	-	92.9%	-	95.7%	-
	PointASNL [61]	Coordinates+Normals	-	93.2%	-	95.9%	-
Convolution-based Methods	Pointwise-CNN [76]	Coordinates	-	86.1%	81.4%	-	-
	PointConv [67]	Coordinates+Normals	-	92.5%	-	-	-
	MC Convolution [68]	Coordinates	-	90.9%	-	-	-
	SpiderCNN [69]	Coordinates+Normals	-	92.4%	-	-	-
	PointCNN [79]	Coordinates	0.45	92.2%	88.1%	-	-
	Flex-Convolution [75]	Coordinates	-	90.2%	-	-	-
	PCNN [70]	Coordinates	1.4	92.3%	-	94.9%	-
	Boulch [63]	Coordinates	-	91.6%	88.1%	-	-
	RS-CNN [62]	Coordinates	-	93.6%	-	-	-
	Spherical CNNs [71]	Coordinates	0.5	88.9%	-	-	-
	GeoCNN [78]	Coordinates	-	93.4%	91.1%	-	-
	Ψ-CNN [77]	Coordinates	-	92.0%	88.7%	94.6%	94.4%
	A-CNN [82]	Coordinates	-	92.6%	90.3%	95.5%	95.3%
	SFCNN [84]	Coordinates	-	91.4%	-	-	-
	SFCNN [84]	Coordinates+Normals	-	92.3%	-	-	-
	DensePoint [64]	Coordinates	0.53	93.2%	-	96.6%	-
	KPConv rigid [65]	Coordinates	-	92.9%	-	-	-
	KPConv deform [65]	Coordinates	-	92.7%	-	-	-
	InterpCNN [80]	Coordinates	12.8	93.0%	-	-	-
	ConvPoint [66]	Coordinates	-	91.8%	88.5%	-	-
Graph-based Methods	ECC [85]	Coordinates	-	87.4%	83.2%	90.8%	90.0%
	KCNet [93]	Coordinates	0.9	91.0%	-	94.4%	-
	DGCNN [87]	Coordinates	1.84	92.2%	90.2%	-	-
	LocalSpecGCN [103]	Coordinates+Normals	-	92.1%	-	-	-
	RGCNN [100]	Coordinates+Normals	2.24	90.5%	87.3%	-	-
	LDGCNN [88]	Coordinates	-	92.9%	90.3%	-	-
	3DTI-Net [105]	Coordinates	2.6	91.7%	-	-	-
	PointGCN [104]	Coordinates	-	89.5%	86.1%	91.9%	91.6%
	ClusterNet [95]	Coordinates	-	87.1%	-	-	-
	Hassani et al. [91]	Coordinates	-	89.1%	-	-	-
	DPAM [92]	Coordinates	-	91.9%	89.9%	94.6%	94.3%
	Grid-GCN [97]	Coordinates	-	93.1%	91.3%	97.5%	97.4%
Hierarchical Data Structure -based Methods	KD-Net [106]	Coordinates	2.0	91.8%	88.5%	94.0%	93.5%
	SO-Net [108]	Coordinates	-	90.9%	87.3%	94.1%	93.9%
	SCN [109]	Coordinates	-	90.0%	87.6%	-	-
	A-SCN [109]	Coordinates	-	89.8%	87.4%	-	-
	3DContextNet [107]	Coordinates	-	90.2%	-	-	-
	3DContextNet [107]	Coordinates+Normals	-	91.1%	-	-	-
Other Methods	3DmFV-Net [52]	Coordinates	4.6	91.6%	-	95.2%	-
	PVNet [110]	Coordinates+Views	-	93.2%	-	-	-
	PVRNet [111]	Coordinates+Views	-	93.6%	-	-	-
	3DPointCapsNet [112]	Coordinates	-	89.3%	-	-	-
	DeepRBFNet [113]	Coordinates	3.2	90.2%	87.8%	-	-
	DeepRBFNet [113]	Coordinates+Normals	3.2	92.1%	88.8%	-	-
	Point2Sequences [114]	Coordinates	-	92.6%	90.4%	95.3%	95.1%
	RCNet [115]	Coordinates	-	91.6%	-	94.7%	-
	RCNet-E [115]	Coordinates	-	92.3%	-	95.6%	-

Several methods also learn from both 3D point clouds and 2D images. In PVNet [110], high-level global features extracted from multi-view images are projected into the subspace of point clouds through an embedding network, and fused with point cloud features through a soft attention mask. Finally, a residual connection is employed for fused features and multi-view features to perform shape recognition. Later, PVRNet [111] is further proposed to exploit the relation between a 3D point cloud and its multiple views by a relation score module. Based on the relation scores, the original 2D global view features are enhanced for point-single-view fusion and point-multi-view fusion.

3.4 Summary

The ModelNet10/40 [6] datasets are the most frequently used datasets for 3D shape classification. Table 2 shows the results achieved by different point-based networks. Several observations can be drawn:

- Pointwise MLP networks are usually served as the basic building block for other types of networks to learn pointwise features.
- As a standard deep learning architecture, convolution-based networks can achieve superior performance on irregular 3D point clouds. More attention should be paid to both discrete and continuous convolution networks for irregular data.

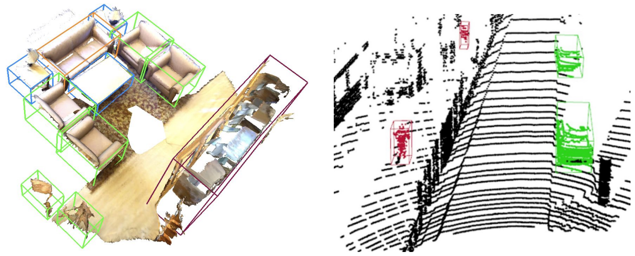
- Due to its inherent strong capability to handle irregular data, graph-based networks have attracted increasingly more attention in recent years. However, it is still challenging to extend graph-based networks in the spectral domain to various graph structures.

4 3D Object Detection and Tracking

In this section, we will review existing methods for 3D object detection, 3D object tracking and 3D scene flow estimation.

4.1 3D Object Detection

A typical 3D object detector takes the point cloud of a scene as its input and produces an oriented 3D bounding box around each detected object, as shown in Fig. 6. Similar to object detection in images [123], 3D object detection methods can be divided into two categories: region proposal-based and single shot methods. Several milestone methods are presented in Fig. 7.



(a) ScanNetV2 [11] dataset

(b) KITTI [14] dataset

Fig. 6: An illustration of 3D object detection. (a) and (b) are originally shown in [124] and [125], respectively.

4.1.1 Region Proposal-based Methods

These methods first propose several possible regions (also called proposals) containing objects, and then extract region-wise features to determine the category label of each proposal. According to their object proposal generation approach, these methods can further be divided into three categories: multi-view based, segmentation-based and frustum-based methods.

Multi-view based Methods. These methods fuse proposal-wise features from different view maps (e.g., LiDAR front view, Bird's Eye View (BEV), and image) to obtain 3D rotated boxes, as shown in Fig. 8(a). The computational cost of these methods is usually high.

Chen et al. [4] generated a group of highly accurate 3D candidate boxes from the BEV map and projected them to the feature maps of multiple views (e.g., LiDAR front view image, RGB image). They then combined these region-wise features from different views to predict oriented 3D bounding boxes, as shown in Fig. 8(a). Although this method achieves a recall of 99.1% at an Intersection over Union (IoU) of 0.25 with only 300 proposals, its speed is too slow for practical applications. Subsequently, several approaches have been developed to improve multi-view 3D object detection methods from two aspects.

First, several methods have been proposed to efficiently fuse the information of different modalities. To generate 3D proposals with a high recall for small objects, Ku et al. [126] proposed a multi-modal fusion-based region proposal network. They first extracted equal-sized features from both BEV and image views using cropping and resizing operations, and then fused these features using element-wise mean pooling. Liang et al. [127] exploited continuous convolutions to enable effective fusion of image and 3D LiDAR feature maps at different resolutions. Specifically, they extracted nearest corresponding image features for each point in the BEV space and then used bilinear interpolation to obtain a dense BEV feature map by projecting image features into the BEV plane. Experimental results show that dense BEV feature maps are more suitable for 3D object detection than discrete image feature maps and sparse LiDAR feature maps. Liang et al. [128] presented a multi-task multi-sensor 3D object detection network for end-to-end training. Specifically, multiple tasks (e.g., 2D object detection, ground estimation and depth completion) are exploited to help the network learn better feature representations. The learned cross-modality representation is further exploited to produce highly accurate object detection results. Experimental results show that this method achieves a significant improvement on 2D, 3D and BEV detection tasks, and outperforms previous state-of-the-art methods on the TOR4D benchmark [129], [130].

Second, different methods have been investigated to extract robust representations of the input data. Lu et al. [39] explored multi-scale contextual information by introducing a Spatial Channel Attention (SCA) module, which captures the global and multi-scale context of a scene and highlights useful features. They also proposed an Extension Spatial Unsample (ESU) module to obtain high-level features with rich spatial information by combining multi-scale low-level features, thus generating reliable 3D object proposals. Although better detection performance can be achieved, the aforementioned multi-view methods take a long runtime since they perform feature pooling for each proposal. Subsequently, Zeng et al. [131] used a pre-RoI pooling convolution to improve the efficiency of [4]. Specifically, they moved the majority of convolution operations to be ahead of the RoI pooling module. Therefore, RoI convolutions are performed once for all object proposals. Experimental results show that this method can run at a speed of 11.1 fps, which is 5 times faster than MV3D [4].

Segmentation-based Methods. These methods first leverage existing semantic segmentation techniques to remove most background points, and then generate a large amount of high-quality proposals on foreground points to save computation, as shown in Fig. 8(b). Compared to multi-view methods [4], [126], [131], these methods achieve higher object recall rates and are more suitable for complicated scenes with highly occluded and crowded objects.

Yang et al. [132] used a 2D segmentation network to predict foreground pixels and projected them into point clouds to remove most background points. They then generated proposals on the predicted foreground points and designed a new criterion named PointsIoU to reduce the redundancy and ambiguity of proposals. Following [132], Shi et al. [133] proposed a PointRCNN framework. Specifically, they directly segmented 3D point clouds to obtain foreground points and then fused semantic features and local spatial

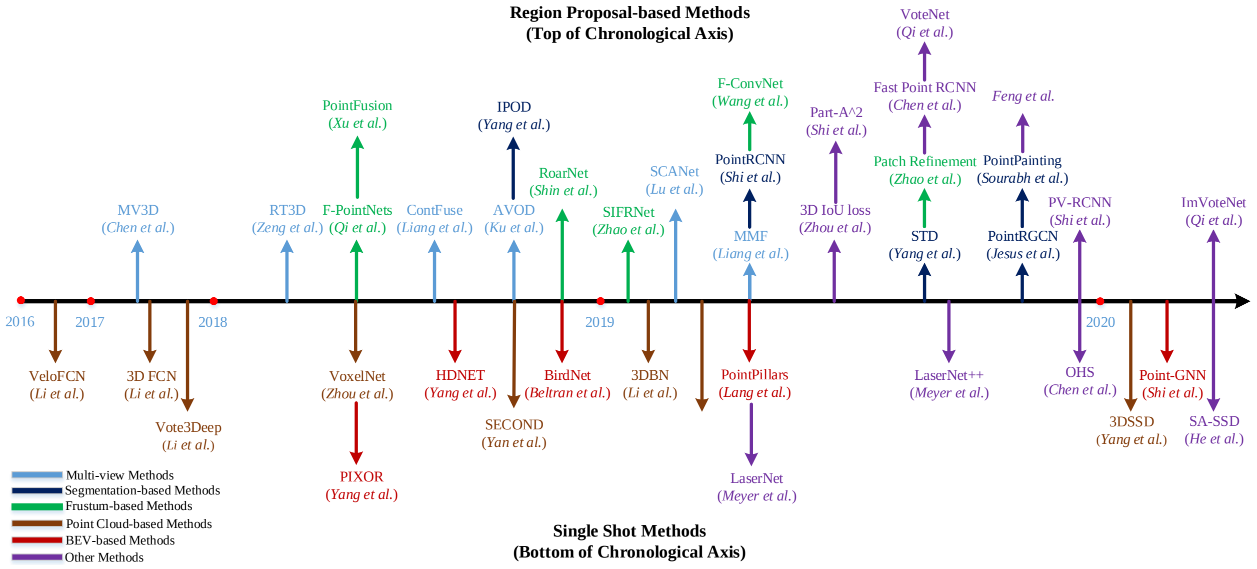


Fig. 7: Chronological overview of the most relevant deep learning-based 3D object detection methods.

features to produce high-quality 3D boxes. Following the Region Proposal Network (RPN) stage of [133], Jesus et al. [134] proposed a pioneering work to leverage Graph Convolution Network (GCN) for 3D object detection. Specifically, two modules are introduced to refine object proposals using graph convolution. The first module R-GCN utilizes all points contained in a proposal to achieve per-proposal feature aggregation. The second module C-GCN fuses per-frame information from all proposals to regress accurate object boxes by exploiting contexts. Sourabh et al. [135] projected a point cloud into the output of the image-based segmentation network and appended the semantic prediction scores to the points. The painted points are fed into existing detectors [133], [136], [137] to achieve significant performance improvement. Yang et al. [138] associated each point with a spherical anchor. The semantic score of each point is then used to remove redundant anchors. Consequently, this method achieves a higher recall with lower computational cost as compared to previous methods [132], [133]. In addition, a PointsPool layer is proposed to learn compact features for interior points in proposals and a parallel IoU branch is introduced to improve localization accuracy and detection performance.

Frustum-based Methods. These methods first leverage existing 2D object detectors to generate 2D candidate regions of objects and then extract a 3D frustum proposal for each 2D candidate region, as shown in Fig. 8(c). Although these methods can efficiently propose possible locations of 3D objects, the step-by-step pipeline makes their performance limited by 2D image detectors.

F-PointNets [139] is a pioneering work in this direction. It generates a frustum proposal for each 2D region and applies PointNet [5] (or PointNet++ [54]) to learn point cloud features of each 3D frustum for amodal 3D box estimation. In a follow-up work, Zhao et al. [140] proposed a Point-SENet module to predict a set of scaling factors, which were further used to adaptively highlight useful features and suppress informative-less features. They also integrated the PointSIFT [141] module into the network to capture orientation information of point clouds, which achieved strong robustness to shape scaling. This method achieves significant improvement on both indoor and outdoor datasets [14], [25] as compared to F-PointNets [139].

Xu et al. [142] leveraged both 2D image region and its corresponding frustum points to accurately regress 3D boxes. To fuse image features and global features of point clouds, they presented a global fusion network for direct regression of box corner locations. They also proposed a dense fusion network for the prediction of point-wise offsets to each corner. Shin et al. [143] first estimated 2D bounding boxes and 3D poses of objects from a 2D image, and then extracted multiple geometrically feasible object candidates. These 3D candidates are fed into a box regression network to predict accurate 3D object boxes. Wang et al. [144] generated a sequence of frustums along the frustum axis for each 2D region and applied PointNet [5] to extract features for each frustum. The frustum-level features are reformed to generate a 2D feature map, which is then fed into a fully convolutional network for 3D box estimation. This method achieves the state-of-the-art performance among 2D image-based methods and was ranked in the top position of the official KITTI leaderboard. Johannes et al. [145] first obtained a preliminary detection results on the BEV map, and then extracted small point subsets (also called patches) based on the BEV predictions. A local refinement network is applied to learn the local features of patches to predict highly accurate 3D bounding boxes.

Other Methods. Motivated by the success of axis-aligned IoU in object detection in images, Zhou et al. [146] integrated the IoU of two 3D rotated bounding boxes into several state-of-the-art detectors [133], [137], [158] to achieve consistent performance improvement. Chen et al. [147] proposed a two-stage network architecture to use both point cloud and voxel representations. First, point clouds are voxelized and fed to a 3D backbone network to produce initial detection results. Second, the interior point features of initial predictions are further exploited for box refinements. Although this design is conceptually simple, it achieves comparable

TABLE 3: Comparative 3D object detection results on the KITTI test 3D detection benchmark. 3D bounding box IoU threshold is 0.7 for cars and 0.5 for pedestrians and cyclists. The modalities are LiDAR (L) and image (I). 'E', 'M' and 'H' represent easy, moderate and hard classes of objects, respectively. For simplicity, we omit the '%' after the value. The symbol '-' means the results are unavailable.

Method			Modality	Speed (fps)	Cars			Pedestrians			Cyclists		
					E	M	H	E	M	H	E	M	H
Region Proposal -based Methods	Multi-view Methods	MV3D [4]	L & I	2.8	74.97	63.63	54.00	-	-	-	-	-	-
		AVOD [126]	L & I	12.5	76.39	66.47	60.23	36.10	27.86	25.76	57.19	42.08	38.29
		ContFuse [127]	L & I	16.7	83.68	68.78	61.67	-	-	-	-	-	-
		MMF [128]	L & I	12.5	88.40	77.43	70.22	-	-	-	-	-	-
		SCANet [39]	L & I	11.1	79.22	67.13	60.65	-	-	-	-	-	-
		RT3D [131]	L & I	11.1	23.74	19.14	18.86	-	-	-	-	-	-
	Segmentation -based Methods	IPOD [132]	L & I	5.0	80.30	73.04	68.73	55.07	44.37	40.05	71.99	52.23	46.50
		PointRCNN [133]	L	10.0	86.96	75.64	70.70	47.98	39.37	36.01	74.96	58.82	52.53
		PointRGCN [134]	L	3.8	85.97	75.73	70.60	-	-	-	-	-	-
		PointPainting [135]	L & I	2.5	82.11	71.70	67.08	50.32	40.97	37.87	77.63	63.78	55.89
		STD [138]	L	12.5	87.95	79.71	75.09	53.29	42.47	38.35	78.69	61.59	55.30
	Frustum -based Methods	F-PointNets [139]	L & I	5.9	82.19	69.79	60.59	50.53	42.15	38.08	72.27	56.12	49.01
		SIFRNet [140]	L & I	-	-	-	-	-	-	-	-	-	-
		PointFusion [142]	L & I	-	77.92	63.00	53.27	33.36	28.04	23.38	49.34	29.42	26.98
		RoarNet [143]	L & I	10.0	83.71	73.04	59.16	-	-	-	-	-	-
		F-ConvNet [144]	L & I	2.1	87.36	76.39	66.69	52.16	43.38	38.80	81.98	65.07	56.54
		Patch Refinement [145]	L	6.7	88.67	77.20	71.82	-	-	-	-	-	-
	Other Methods	3D IoU loss [146]	L	12.5	86.16	76.50	71.39	-	-	-	-	-	-
		Fast Point R-CNN [147]	L	16.7	84.80	74.59	67.27	-	-	-	-	-	-
		PV-RCNN [148]	L	12.5	90.25	81.43	76.82	-	-	-	-	-	-
		VoteNet [124]	L	-	-	-	-	-	-	-	-	-	-
		Feng et al. [149]	L	-	-	-	-	-	-	-	-	-	-
		ImVoteNet [150]	L & I	-	-	-	-	-	-	-	-	-	-
		Part-A^2 [151]	L	12.5	87.81	78.49	73.51	-	-	-	-	-	-
Single Shot Methods	BEV-based Methods	PIXOR [129]	L	28.6	-	-	-	-	-	-	-	-	-
		HDNET [152]	L	20.0	-	-	-	-	-	-	-	-	-
		BirdNet [153]	L	9.1	13.53	9.47	8.49	12.25	8.99	8.06	16.63	10.46	9.53
	Discretization -based Methods	VeloFCN [154]	L	1.0	-	-	-	-	-	-	-	-	-
		3D FCN [155]	L	<0.2	-	-	-	-	-	-	-	-	-
		Vote3Deep [156]	L	-	-	-	-	-	-	-	-	-	-
		3DBN [157]	L	7.7	83.77	73.53	66.23	-	-	-	-	-	-
		VoxelNet [136]	L	2.0	77.47	65.11	57.73	39.48	33.69	31.51	61.22	48.36	44.37
		SECOND [158]	L	26.3	83.34	72.55	65.82	48.96	38.78	34.91	71.33	52.08	45.83
		MVX-Net [159]	L & I	16.7	84.99	71.95	64.88	-	-	-	-	-	-
		PointPillars [137]	L	62.0	82.58	74.31	68.99	51.45	41.92	38.89	77.10	58.65	51.92
		SA-SSD [160]	L	25.0	88.75	79.79	74.16	-	-	-	-	-	-
	Point-based Methods	3DSSD [161]	L	25.0	88.36	79.57	74.55	54.64	44.27	40.23	82.48	64.10	56.90
	Other Methods	LaserNet [162]	L	83.3	-	-	-	-	-	-	-	-	-
		LaserNet++ [163]	L & I	26.3	-	-	-	-	-	-	-	-	-
		OHS-Dense [164]	L	33.3	88.12	78.34	73.49	47.14	39.72	37.25	79.09	62.72	56.76
		OHS-Direct [164]	L	33.3	86.40	77.74	72.97	51.29	44.81	41.13	77.70	63.16	57.16
		Point-GNN [125]	L	1.7	88.33	79.47	72.29	51.92	43.77	40.14	78.60	63.48	57.08

performance to [133] while maintaining a speed of 16.7 fps. Shi et al. [148] proposed PointVoxel-RCNN (PV-RCNN) to leverage both 3D convolutional network and PointNet-based set abstraction for the learning of point cloud features. Specifically, the input point clouds are first voxelized and then fed into a 3D sparse convolutional network to generate high-quality proposals. The learned voxel-wise features are then encoded into a small set of key points via a voxel set abstraction module. In addition, they also proposed a keypoint-to-grid ROI abstraction module to capture rich context information for box refinement. Experimental results show that this method outperforms previous methods by a remarkable margin and is ranked first[1] on the Car class of the KITTI 3D detection benchmark.

Inspired by Hough voting-based 2D object detectors, Qi et al. [124] proposed VoteNet to directly vote for virtual center points of objects from point clouds and to generate a group of high-quality 3D object proposals by aggregating vote features. VoteNet significantly outperforms previous approaches using only geometric information, and achieves the state-of-the-art performance on two large indoor benchmarks (i.e., ScanNet [11] and SUN RGB-D [25]). However, the prediction of virtual center point is unstable for a partially occluded object. Further, Feng et al. [149] added an auxiliary branch of direction vectors to improve the prediction accuracy of virtual center points and 3D candidate boxes. In addition, a 3D object-object relationship graph between proposals is built to emphasize useful features for accurate object detection. Qi et al. [150] proposed an ImVoteNet detector by fusing 2D object detection cues (e.g., geometric and semantic/texture cues) into a 3D voting pipeline. Inspired by the observation that the ground truth boxes of 3D objects provide accurate locations of intra-object parts, Shi et al. [151] proposed the Part-A^2 Net, which is composed of a part-aware stage and a part-aggregation stage. The part-aware stage applies a UNet-like [165] network with sparse convolution and sparse deconvolution to learn point-wise features for the prediction and coarse generation of intra-object part locations. The part-aggregation stage adopts RoI-aware pooling to aggregate predicted part locations for box refinement.

[1] The ranking refers to the time of the submission: 12th June, 2020

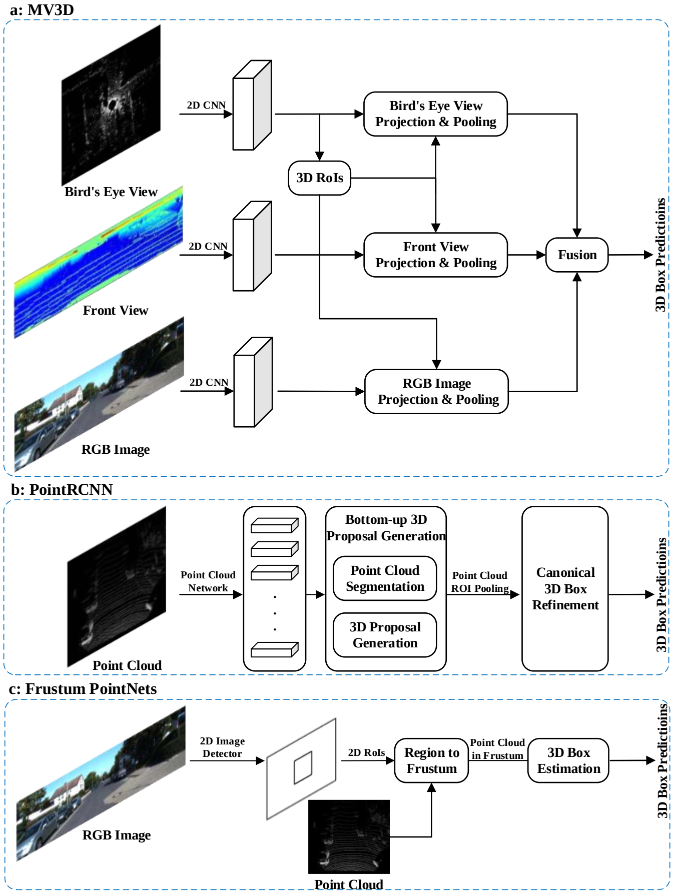


Fig. 8: Typical networks for three categories of region proposal-based 3D object detection methods. From top to bottom: (a) multi-view based, (b) segmentation-based and (c) frustum-based methods.

4.1.2 Single Shot Methods

These methods directly predict class probabilities and regress 3D bounding boxes of objects using a single-stage network. They do not need region proposal generation and post-processing. As a result, they can run at a high speed. According to the type of input data, single shot methods can be divided into three categories: BEV-based, discretization-based and point-based methods.

BEV-based Methods. These methods mainly take BEV representation as their input. Yang et al. [129] discretized the point cloud of a scene with equally spaced cells and encoded the reflectance in a similar way, resulting in a regular representation. A Fully Convolution Network (FCN) network is then applied to estimate the locations and heading angles of objects. This method outperforms most single shot methods (including VeloFCN [154], 3D-FCN [155] and Vote3Deep [156]) while running at 28.6 fps. Later, Yang et al. [152] exploited the geometric and semantic prior information provided by High-Definition (HD) maps to improve the robustness and detection performance of [129]. Specifically, they obtained the coordinates of ground points from the HD map and then used the distance relative to the ground for BEV representation to remedy the translation variance caused by the slope of the road. In addition, they concatenated a binary road mask with the BEV representation along the channel dimension to focus on moving objects. Since HD maps are not available everywhere, they also proposed an online map prediction module to estimate the map priors from single LiDAR point cloud. This map-aware method significantly outperforms its baseline on the TOR4D [129], [130] and KITTI [14] datasets. However, its generalization performance to point clouds with different densities is poor. To solve this problem, Beltrán et al. [153] proposed a normalization map to consider the differences among different LiDAR sensors. The normalization map is a 2D grid with the same resolution as the BEV map, and it encodes the maximum number of points contained in each cell. It is shown that this normalization map significantly improves the generalization ability of BEV-based detectors.

Discretization-based Methods. These methods convert a point cloud into a regular discrete representation, and then apply CNN to predict both categories and 3D boxes of objects.

Li et al. [154] proposed the first method to use a FCN for 3D object detection. They converted a point cloud into a 2D point map and used a 2D FCN to predict the bounding boxes and confidences of objects. Later, they [155] discretized the point cloud into a 4D tensor with dimensions of length, width, height and channels, and extended the 2D FCN-based detection technologies to 3D domain for 3D object detection. Compared to [154], 3D FCN-based method [155] obtains a gain of over 20% in accuracy, but inevitably costs more computing resources due to 3D convolutions and the sparsity of the data. To address the sparsity problem of voxels, Engelcke et al. [156] leveraged a feature-centric voting scheme to generate a set of votes for each non-empty voxel and to obtain the convolutional results by accumulating the votes. Its computational complexity is proportional to the number of occupied voxels. Li et al. [157] constructed a 3D backbone network by stacking multiple sparse 3D CNNs. This method is designed to save memory and accelerate computation by fully using the sparsity of voxels. This 3D backbone network extracts rich 3D features for object detection without introducing heavy computational burden.

Zhou et al. [136] presented a voxel-based end-to-end trainable framework VoxelNet. They partitioned a point cloud into equally spaced voxels and encoded the features within each voxel into a 4D tensor. A region proposal network is then connected to produce detection results. Although its performance is strong, this method is very slow due to the sparsity of voxels and 3D convolutions. Later, Yan et al. [158] used the sparse convolutional network [166] to improve the inference efficiency of [136]. They also proposed a sine-error angle loss to solve the ambiguity between orientations of 0 and π. Sindagi et al. [159] extended VoxelNet by fusing image and point cloud features at early stages. Specifically, they projected non-empty voxels generated by [136] into the image and used a pre-trained network to extract image features for each projected voxel. These image features are then concatenated with voxel features to produce accurate 3D boxes. Compared to [136], [158], this method can effectively exploit multi-modal information to reduce false positives and negatives. Lang et al. [137] proposed a 3D object detector named PointPillars. This method leverages PointNet [5] to learn the feature of point clouds organized in vertical columns (Pillars) and encodes the learned features as a pesudo image. A 2D object detection pipeline is then applied to predict 3D bounding boxes. PointPillars outperforms most fusion approaches (including MV3D [4], RoarNet [143] and AVOD [126]) in terms of Average Precision (AP). Moreover, PointPillars can run at a speed of 62 fps on both the 3D and BEV KITTI [14] benchmarks, making it highly suitable for practical applications.

Inspired by the observation that partial spatial information of a point cloud is inevitably lost in progressively downscaled feature maps of existing single shot detectors, He et al. [160] proposed a SA-SSD detector to leverage the fine-grained structure information to improve localization accuracy. Specifically, they first converted a point cloud to a tensor and fed it into a backbone network to extract multi-stage features. In addition, an auxiliary network with point-level supervision is employed to guide the features to learn the structure of point clouds. Experimental results show that SA-SSD ranks the first[2] on the Car class of the KITTI BEV detection benchmark.

Point-based Methods. These methods directly take raw point clouds as their inputs. 3DSSD [161] is a pioneering work in this direction. It introduces a fusion sampling strategy for Distance-FPS (D-FPS) and Feature-FPS (F-FPS) to remove time-consuming Feature Propagation (FP) layers and the refinement module in [133]. Then, a Candidate Generation (CG) layer is used to fully exploit representative points, which are further fed into an anchor-free regression head with a 3D centerness label to predict 3D object boxes. Experimental results show that 3DSSD outperforms the two-stage point-based method PointRCNN [133] while maintaining a speed of 25 fps.

Other Methods. Meyer et al. [162] proposed an efficient 3D object detector called LaserNet. This method predicts a probability distribution over bounding boxes for each point and then combines these per-point distributions to generate final 3D object boxes. Further, the dense Range View (RV) representation of point cloud is used as input and a fast mean-shift algorithm is proposed to reduce the noise produced by per-point prediction. LaserNet achieves the state-of-the-art performance at the range of 0 to 50 meters, and its runtime is significantly lower than existing methods. Meyer et al. [163] then extended LaserNet [162] to exploit the dense texture provided by RGB images (e.g., 50 to 70 meters). Specifically, they associated LiDAR points with image pixels by projecting 3D point clouds onto 2D images and exploited this association to fuse RGB information into 3D points. They also considered 3D semantic segmentation as an auxiliary task to learn better representations. This method achieves a significant improvement in both long-range (e.g., 50 to 70 meters) object detection and semantic segmentation while maintaining high efficiency of LaserNet. Inspired by the observation that points on an isolated object part can provide abundant information about position and orientation of the object, Chen et al. [164] proposed a novel $Hotspot$ representation and the first hotspot-based anchor-free detector. Specifically, raw point clouds are first voxelized and then fed into a backbone network to produce 3D feature maps. These feature maps are used to classify hotspots and predict 3D bounding boxes simultaneously. Note that, hotspots are assigned at the last convolutional layer of the backbone network. Experimental results show that this method achieves comparable performance and is robust to sparse point clouds. Shi et el. [125] proposed a graph neural network Point-GNN to detect 3D objects from lidar point clouds. They first encoded an input point cloud as a graph of near neighbors with a fixed radius and then fed the graph into Point-GNN to predict both the categories and boxes of objects.

4.2 3D Object Tracking

Given the locations of an object in the first frame, the task of object tracking is to estimate its state in subsequent frames [167], [168]. Since 3D object tracking can use the rich geometric information in point clouds, it is expected to overcome several drawbacks faced by image-based tracking, including occlusion, illumination and scale variation.

Inspired by the success of Siamese network [169] for imaged-based object tracking, Giancola et al. [170] proposed a 3D Siamese network with shape completion regularization. Specifically, they first generated candidates using a Kalman filter, and encoded model and candidates into a compact representation using shape regularization. The cosine similarity is then used to search the location of the tracked object in the next frame. This method can be used as an alternative for object tracking, and significantly outperforms most 2D object tracking methods, including $\mathrm{STAPLE_{CA}}$ [171] and SiamFC [169]. To efficiently search the target object, Zarzar et al. [172] leveraged a 2D Siamese network to generate a large number of coarse object candidates on BEV representation. They then refined the candidates by exploiting the cosine similarity in 3D Siamese network. This method significantly outperforms [170] in terms of both precision (i.e., by 18%) and success rate (i.e., by 12%). Simon et al. [173] proposed a 3D object detection and tracking architecture for semantic point clouds. They first generated voxelized semantic point clouds by fusing 2D visual semantic information, and then utilized the temporal information to improve accuracy and robustness of multi-target tracking. In addition, they introduced a powerful and simplified evaluation metric (i.e., Scale-Rotation-Translation score (SRFs)) to speed up training and inference. Complexer-YOLO achieves promising tracking performance and can still run in real-time. Further, Qi et al. [174] proposed a Point-to-Box (P2B) network. They fed template and search areas into the backbone to obtain their seeds. The search area seeds are augmented with target-specific features and then the potential target centers are regressed by Hough voting. Experimental results show that P2B outperforms [170] by over 10% while running at 40 fps.

4.3 3D Scene Flow Estimation

Given two point clouds $\mathcal{X}$ and $\mathcal{Y}$, 3D scene flow $D = \{d_i\}^N$ describes the movement of each point x_i in $\mathcal{X}$ to its corresponding position x_i' in $\mathcal{Y}$, such that $x_i' = x_i + d_i$. Figure 9 shows a 3D scene flow between two KITTI point clouds. Analogous to optical flow estimation in 2D vision, several methods have started to learn useful information (e.g. 3D scene flow, spatial-temporary information) from a sequence of point clouds.

[2] The ranking refers to the time of the submission: 12th June, 2020

TABLE 4: Comparative 3D object detection results on the KITTI test BEV detection benchmark. 3D bounding box IoU threshold is 0.7 for cars and 0.5 for pedestrians and cyclists. The modalities are LiDAR (L) and image (I). 'E', 'M' and 'H' represent easy, moderate and hard classes of objects, respectively. For simplicity, we omit the '%' after the value. The symbol '-' means the results are unavailable.

Method			Modality	Speed (fps)	Cars			Pedestrians			Cyclists		
					E	M	H	E	M	H	E	M	H
Region Proposal -based Methods	Multi-view Methods	MV3D [4]	L & I	2.8	86.62	78.93	69.80	-	-	-	-	-	-
		AVOD [126]	L & I	12.5	89.75	84.95	78.32	42.58	33.57	30.14	64.11	48.15	42.37
		ContFuse [127]	L & I	16.7	94.07	85.35	75.88	-	-	-	-	-	-
		MMF [128]	L & I	12.5	93.67	88.21	81.99	-	-	-	-	-	-
		SCANet [39]	L & I	11.1	90.33	82.85	76.06	-	-	-	-	-	-
		RT3D [131]	L & I	11.1	56.44	44.00	42.34	-	-	-	-	-	-
	Segmentation -based Methods	IPOD [132]	L & I	5.0	89.64	84.62	79.96	60.88	49.79	45.43	78.19	59.40	51.38
		PointRCNN [133]	L	10.0	92.13	87.39	82.72	54.77	46.13	42.84	82.56	67.24	60.28
		PointRGCN [134]	L	3.8	91.63	87.49	80.73	-	-	-	-	-	-
		PointPainting [135]	L & I	2.5	92.45	88.11	83.36	58.70	49.93	46.29	83.91	71.54	62.97
		STD [138]	L	12.5	94.74	89.19	86.42	60.02	48.72 44.55	81.36	67.23	59.35	
	Frustum -based Methods	F-PointNets [139]	L & I	5.9	91.17	84.67	74.77	57.13	49.57	45.48	77.26	61.37	53.78
		SIFRNet [140]	L & I	-	-	-	-	-	-	-	-	-	-
		PointFusion [142]	L & I	-	-	-	-	-	-	-	-	-	-
		RoarNet [143]	L & I	10.0	88.20	79.41	70.02	-	-	-	-	-	-
		F-ConvNet [144]	L & I	2.1	91.51	85.84	76.11	57.04	48.96	44.33	84.16	68.88	60.05
		Patch Refinement [145]	L	6.7	92.72	88.39	83.19	-	-	-	-	-	-
	Other Methods	3D IoU loss [146]	L	12.5	91.36	86.22	81.20	-	-	-	-	-	-
		Fast Point R-CNN [147]	L	16.7	90.76	85.61	79.99	-	-	-	-	-	-
		PV-RCNN [148]	L	12.5	94.98	90.65	86.14	-	-	-	82.49	68.89	62.41
		VoteNet [124]	L	-	-	-	-	-	-	-	-	-	-
		Feng et al. [149]	L	-	-	-	-	-	-	-	-	-	-
		ImVoteNet [150]	L & I	-	-	-	-	-	-	-	-	-	-
		Part-A^2 [151]	L	12.5	91.70	87.79	84.61	-	-	-	81.91	68.12	61.92
Single Shot Methods	BEV-based Methods	PIXOR [129]	L	28.6	83.97	80.01	74.31	-	-	-	-	-	-
		HDNET [152]	L	20.0	89.14	86.57	78.32	-	-	-	-	-	-
		BirdNet [153]	L	9.1	76.88	51.51	50.27	20.73	15.80	14.59	36.01	23.78	21.09
	Discretization -based Methods	VeloFCN [154]	L	1.0	0.02	0.14	0.21	-	-	-	-	-	-
		3D FCN [155]	L	<0.2	70.62	61.67	55.61	-	-	-	-	-	-
		Vote3Deep [156]	L	-	-	-	-	-	-	-	-	-	-
		3DBN [157]	L	7.7	89.66	83.94	76.50	-	-	-	-	-	-
		VoxelNet [136]	L	2.0	89.35	79.26	77.39	46.13	40.74	38.11	66.70	54.76	50.55
		SECOND [158]	L	26.3	89.39	83.77	78.59	55.99	45.02	40.93	76.50	56.05	49.45
		MVX-Net [159]	L & I	16.7	92.13	86.05	78.68	-	-	-	-	-	-
		PointPillars [137]	L	62.0	90.07	86.56	82.81	57.60	48.64	45.78	79.90	62.73	55.58
		SA-SSD [160]	L	25.0	95.03	91.03	85.96	-	-	-	-	-	-
	Point-based Methods	3DSSD [161]	L	25.0	92.66	89.02	85.86	60.54	49.94	45.73	85.04	67.62	61.14
	Other Methods	LaserNet [162]	L	83.3	79.19	74.52	68.45	-	-	-	-	-	-
		LaserNet++ [163]	L & I	26.3	-	-	-	-	-	-	-	-	-
		OHS-Dense [164]	L	33.3	93.73	88.11	84.98	50.87	44.59	42.14	82.13	66.86	60.86
		OHS-Direct [164]	L	33.3	93.59	87.95	83.21	55.90	49.48	45.79	79.66	67.20	61.04
		Point-GNN [125]	L	1.7	93.11	89.17	83.90	55.36	47.07	44.61	81.17	67.28	59.67

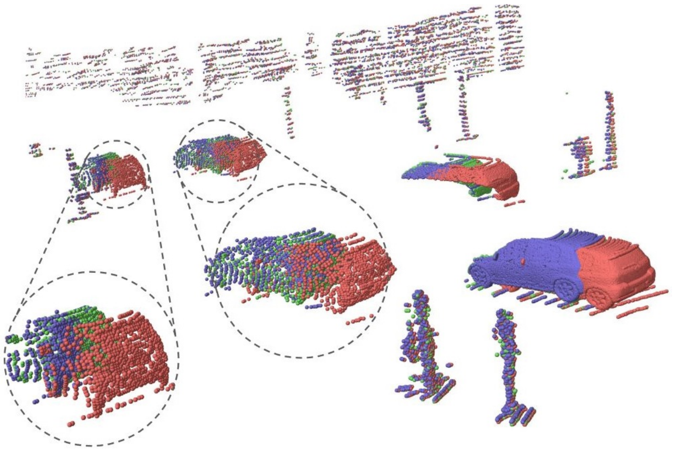


Fig. 9: A 3D scene flow between two KITTI point clouds, originally shown in [175]. Point clouds $\mathcal{X}$, $\mathcal{Y}$ and the translated point cloud of $\mathcal{X}$ are highlighted in red, green, and blue, respectively.

Liu et al. [175] proposed FlowNet3D to directly learn scene flows from a pair of consecutive point clouds. FlowNet3D learns both point-level features and motion features through a flow embedding layer. However, there are two problems with FlowNet3D. First, some predicted motion vectors differ significantly from the ground truth in their directions. Second, it is difficult to apply FlowNet to non-static scenes, especially for the scenes which are dominated by deformable objects. To solve this problem, Wang et al. [176] introduced a cosine distance loss to minimize the angle between the predictions and the ground truth. In addition, they also proposed a point-to-plane distance loss to improve the accuracy for both rigid and dynamic scenes. Experimental results show that these two loss terms improve the accuracy of FlowNet3D from 57.85% to 63.43%, and speed up and stabilize the training process. Gu et al. [177] proposed a Hierarchical Permutohedral Lattice FlowNet (HPLFlowNet) to directly estimate scene flow from large-scale point clouds. Several bilateral convolution layers are proposed to restore structural information from raw point clouds, while reducing the computational cost.

To effectively process sequential point clouds, Fan and Yang [178] proposed PointRNN, PointGRU and PointLSTM networks and a sequence-to-sequence model to track moving points. PointRNN, PointGRU, and PointLSTM are able to capture the spatial-temporary information and model dynamic point clouds. Similarly, Liu et al. [179] proposed MeteorNet to directly learn a representation from dynamic point clouds. This method learns to aggregate information from spatiotemporal neighboring points. Direct grouping and chained-flow grouping are further introduced to determine the temporal neighbors. However, the performance of the aforementioned methods is limited by the scale of datasets. Mittal et al. [180] proposed two self-supervised losses to train their network on large unlabeled datasets. Their main idea is that a robust scene flow estimation method should be effective in both forward and backward predictions. Due to the unavailability of scene flow annotation, the nearest neighbor of the predicted transformed point is considered as pesudo ground truth. However, the true ground truth may not be the same as the nearest point. To avoid this problem, they computed the scene flow in the reverse direction and proposed a cycle consistency loss to translate the point to the original position. Experimental results show that this self-supervised method exceeds the state-of-the-art performance of supervised learning-based methods.

4.4 Summary

The KITTI [14] benchmark is one of the most influential datasets in autonomous driving and has been commonly used in both academia and industry. Tables 3 and 4 present the results achieved by different detectors on the KITTI test 3D benchmarks. The following observations can be made:

- Region proposal-based methods are the most frequently investigated methods among these two categories, and outperform single shot methods by a large margin on both KITTI test 3D and BEV benchmarks.
- There are two limitations for existing 3D object detectors. First, the long-range detection capability of existing methods is relatively poor. Second, how to fully exploit the texture information in images is still an open problem.
- Multi-task learning is a future direction in 3D object detection. E.g., MMF [128] learns a cross-modality representation to achieve state-of-the-art detection performance by incorporating multiple tasks.
- 3D object tracking and scene flow estimation are emerging research topics, and have gradually attracted increasing attention since 2019.

5 3D Point Cloud Segmentation

3D point cloud segmentation requires the understanding of both the global geometric structure and the fine-grained details of each point. According to the segmentation granularity, 3D point cloud segmentation methods can be classified into three categories: *semantic segmentation* (scene level), *instance segmentation* (object level) and *part segmentation* (part level).

5.1 3D Semantic Segmentation

Given a point cloud, the goal of semantic segmentation is to separate it into several subsets according to the semantic meanings of points. Similar to the taxonomy for 3D shape classification (Section 3), there are four paradigms for semantic segmentation: projection-based, discretization-based, point-based, and hybrid methods.

The first step of both the projection and discretization-based methods is to transform a point cloud to an intermediate regular representation, such as multi-view [181], [182], spherical [183], [184], [185], volumetric [166], [186], [187], permutohedral lattice [188], [189], and hybrid representations [190], [191], as shown in Fig. 11. The intermediate segmentation results are then projected back to the raw point cloud. In contrast, point-based methods directly work on irregular point clouds. Several representative methods are shown in Fig. 10.

5.1.1 Projection-based Methods

These methods usually project a 3D point cloud into 2D images, including multi-view and spherical images.

Multi-view Representation. Lawin et al. [181] first projected a 3D point cloud onto 2D planes from multiple virtual camera views. Then, a multi-stream FCN is used to predict pixel-wise scores on synthetic images. The final semantic label of each point is obtained by fusing the reprojected scores over different views. Similarly, Boulch et al. [182] first generated several RGB and depth snapshots of a point cloud using multiple camera positions. They then performed pixel-wise labeling on these snapshots using 2D segmentation networks. The scores predicted from RGB and depth images are further fused using residual correction [192]. Based on the assumption that point clouds are sampled from locally Euclidean surfaces, Tatarchenko et al. [193] introduced tangent convolutions for dense point cloud segmentation. This method first projects the local surface geometry around each point to a virtual tangent plane. Tangent convolutions are then directly operated on the surface geometry. This method shows great scalability and is able to process large-scale point clouds with millions of points. Overall, the performance of multi-view segmentation methods is sensitive to viewpoint selection and occlusions. Besides, these methods have not fully exploited the underlying geometric and structural information, as the projection step inevitably introduces information loss.

Spherical Representation. To achieve fast and accurate segmentation of 3D point clouds, Wu et al. [183] proposed an end-to-end network based on SqueezeNet [194] and Conditional Random Field (CRF). To further improve segmentation accuracy, SqueezeSegV2 [184] is introduced to address domain shift by utilizing an unsupervised domain adaptation pipeline. Milioto et al. [185] proposed RangeNet++ for real-time semantic segmentation of LiDAR point clouds. The semantic labels of 2D range images are first transferred to 3D point clouds, an efficient GPU-enabled KNN-based post-processing step is further used to alleviate the problem of discretization errors and blurry inference outputs. Compared to single view projection, spherical projection retains more information and is suitable for the labeling of LiDAR point clouds. However, this intermediate representation

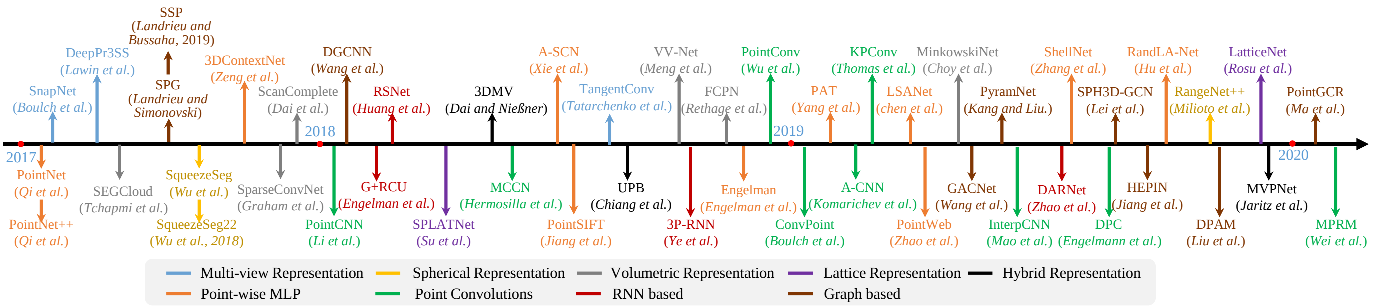
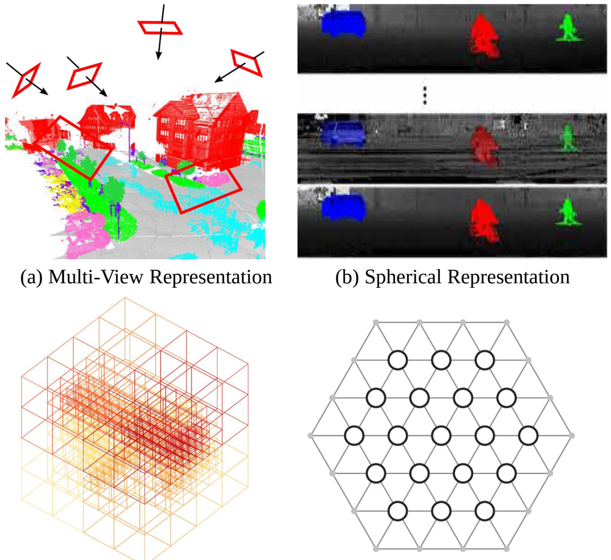


Fig. 10: Chronological overview of the most relevant deep learning-based 3D semantic segmentation methods.



(a) Multi-View Representation (b) Spherical Representation

(c) Dense Discretization Representation (d) Sparse Discretization Representation

Fig. 11: An illustration of the intermediate representation. (a) and (b) are originally shown in [182] and [183], respectively.

inevitably brings several problems such as discretization errors and occlusions.

5.1.2 Discretization-based Methods

These methods usually convert a point cloud into a dense/sparse discrete representation, such as volumetric and sparse permutohedral lattices.

Dense Discretization Representation. Early methods usually voxelized the point clouds as dense grids and then leverage the standard 3D convolutions. Huang et al. [195] first divided a point cloud into a set of occupancy voxels, then fed these intermediate data to a fully-3D CNN for voxel-wise segmentation. Finally, all points within a voxel are assigned the same semantic label as the voxel. The performance of this method is severely limited by the granularity of the voxels and the boundary artifacts caused by the point cloud partition. Further, Tchapmi et al. [196] proposed SEGCloud to achieve fine-grained and global consistent semantic segmentation. This method introduces a deterministic trilinear interpolation to map the coarse voxel predictions generated by 3D-FCNN [197] back to the point cloud, and then uses Fully Connected CRF (FC-CRF) to enforce spatial consistency of these inferred per-point labels. Meng et al. [186] introduced a kernel-based interpolated variational autoencoder architecture to encode the local geometrical structures within each voxel. Instead of a binary occupancy representation, RBFs are employed for each voxel to obtain a continuous representation and capture the distribution of points in each voxel. VAE is further used to map the point distribution within each voxel to a compact latent space. Then, both symmetry groups and an equivalence CNN are used to achieve robust feature learning.

Thanks to the good scalability of 3D CNN, volumetric-based networks are free to be trained and tested on point clouds with different spatial sizes. In Fully-Convolutional Point Network (FCPN) [187], different levels of geometric relations are first hierarchically abstracted from point clouds, 3D convolutions and weighted average pooling are then used to extract features and incorporate long-range dependencies. This method can process large-scale point clouds and has good scalability during inference. Dai et al. [198] proposed ScanComplete to achieve 3D scan completion and per-voxel semantic labeling. This method leverages the scalability of fully-convolutional neural networks and can adapt to different input data sizes during training and test. A coarse-to-fine strategy is used to hierarchically improve the resolution of the predicted results.

Overall, the volumetric representation naturally preserves the neighborhood structure of 3D point clouds. Its regular data format also allows direct application of standard 3D convolutions. These factors lead to a steady performance improvement in this area. However, the voxelization step inherently introduces discretization artifacts and information loss. Usually, a high resolution leads to high memory and computational costs, while a low resolution introduces loss of details. It is non-trivial to select an appropriate grid resolution in practice.

Sparse Discretization Representation. Volumetric representation is naturally sparse, as the number of non-zero values only accounts for a small percentage. Therefore, it is inefficient to apply dense convolution neural networks on the spatially-sparse data. To this end, Graham et al. [166] proposed submanifold sparse convolutional networks based on the indexing structure. This method significantly reduces memory and computational costs by restricting the output of convolution to be only related to occupied voxels. Meanwhile, its sparse convolution can also control the sparsity of the extracted features. This submanifold sparse convolution is suitable for efficient processing of high-dimensional and spatially-sparse data. Further, Choy et al. [199] proposed a 4D spatio-temporal convolutional neural network called MinkowskiNet for 3D video perception. A generalized sparse convolution is proposed to effectively process high-dimensional data. A trilateral-stationary conditional random field is further applied to enforce consistency.

On the other hand, Su et al. [188] proposed the Sparse Lattice Networks (SPLATNet) based on Bilateral Convolution Layers (BCLs). This method first interpolates a raw point cloud to a permutohedral sparse lattice, BCL is then applied to convolve on occupied parts of the sparsely populated lattice. The filtered output is then interpolated back to the raw point cloud. In addition, this method allows flexible joint processing of multi-view images and point clouds. Further, Rosu et al. [189] proposed LatticeNet to achieve efficient processing of large point clouds. A data-dependent interpolation module called DeformsSlice is also introduced to back project the lattice feature to point clouds.

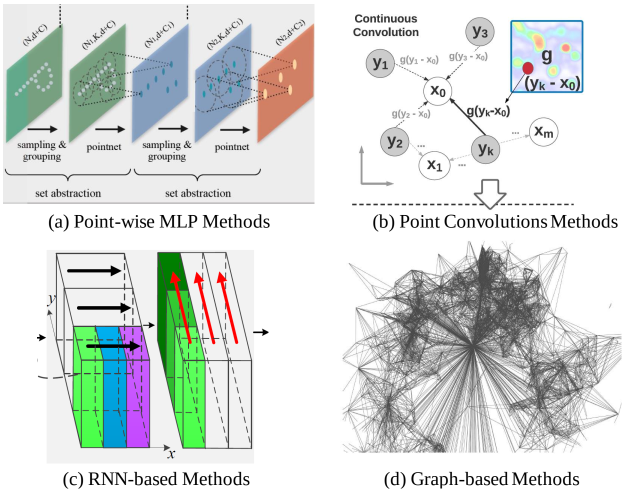
5.1.3 Hybrid Methods

To further leverage all available information, several methods have been proposed to learn multi-modal features from 3D scans. Dai and Nießner [190] presented a joint 3D-multi-view network to combine RGB features and geometric features. A 3D CNN stream and several 2D streams are used to extract features, and a differentiable back-projection layer is proposed to jointly fuse the learned 2D embeddings and 3D geometric features. Further, Chiang et al. [200] proposed a unified point-based framework to learn 2D textural appearance, 3D structures and global context features from point clouds. This method directly applies point-based networks to extracts local geometric features and global context from sparsely sampled point sets without any voxelization. Jaritz et al. [191] proposed Multi-view PointNet (MVPNet) to aggregate appearance features from 2D multi-view images and spatial geometric features in the canonical point cloud space.

5.1.4 Point-based Methods

Point-based networks directly work on irregular point clouds. However, point clouds are orderless and unstructured, making it infeasible to directly apply standard CNNs. To this end, the pioneering work PointNet [5] is proposed to learn per-point features using shared MLPs and global features using symmetrical pooling functions. Based on PointNet, a series of point-based networks have been proposed recently. Overall, these methods can be roughly divided into pointwise MLP methods, point convolution methods, RNN-based methods, and graph-based methods.

Pointwise MLP Methods. These methods usually use shared MLP as the basic unit in their network for its high efficiency. However, point-wise features extracted by shared MLP cannot capture the local geometry in point clouds and the mutual interactions between points [5]. To capture wider context for each point and learn richer local structures, several dedicated networks have been introduced, including methods based on neighboring feature pooling, attention-based aggregation, and local-global feature concatenation.



(a) Point-wise MLP Methods

(b) Point Convolutions Methods

(c) RNN-based Methods

(d) Graph-based Methods

Fig. 12: An illustration of point-based methods. (a)-(d) are originally shown in [54], [201], [202], [203], respectively.

Neighboring feature pooling: To capture local geometric patterns, these methods learn a feature for each point by aggregating the information from local neighboring points. In particular, PointNet++ [54] groups points hierarchically and progressively learns from larger local regions, as illustrated in Fig. 12(a). Multi-scale grouping and multi-resolution grouping are also proposed to overcome the problems caused by non-uniformity and varying density of point clouds. Later, Jiang et al. [141] proposed a PointSIFT module to achieve orientation encoding and scale awareness. This module stacks and encodes the information from eight spatial orientations through a three-stage ordered convolution. Multi-scale features are concatenated to achieve adaptivity to different scales. Different from the grouping techniques used in PointNet++ (i.e., ball query), Engelmann et al. [204] utilized K-means clustering and KNN to separately define two neighborhoods in the world space and feature space. Based on the assumption that points from the same class are expected to be closer in feature space, a pairwise distance loss and a centroid loss are introduced to further regularize feature learning. To model the mutual interactions between different points, Zhao et al. [57] proposed PointWeb to explore the relations between all pairs of points in a local region by densely constructing a locally fully-linked web. An Adaptive Feature Adjustment (AFA) module is proposed to achieve information interchange and feature refinement. This aggregation operation helps the network to learn a discriminative feature representation. Zhang et al. [205] proposed a permutation invariant convolution called Shellconv based on the statistics from concentric spherical shells. This method first queries a set of multi-scale concentric spheres, the max-pooling operation is then used within different shells to summarize the statistics, MLPs and 1D convolution are used to obtain the final convolution output. Hu et al. [206] proposed an efficient and lightweight network called RandLA-Net for large-scale point cloud segmentation. This network utilizes random point sampling to achieve remarkably high efficiency in terms of memory and computation. A local feature aggregation module is further proposed to capture and preserve geometric features.

Attention-based aggregation: To further improve segmentation accuracy, an attention mechanism [120] is introduced to point cloud segmentation. Yang et al. [56] proposed a group shuffle attention to model the relations between points, and presented a permutation-invariant, task-agnostic and differentiable Gumbel Subset Sampling (GSS) to replace the widely used FPS approach. This module is less sensitive to outliers and can select a representative subset of points. To better capture the spatial distribution of a point cloud, Chen et al. [207] proposed a Local Spatial Aware (LSA) layer to learn spatial awareness weights based on the spatial layouts and the local structures of point clouds. Similar to CRF, Zhao et al. [208] proposed an Attention-based Score Refinement (ASR) module to post-process the segmentation results produced by the network. The initial segmentation result is refined by pooling the scores of neighboring points with learned attention weights. This module can be easily integrated into existing deep networks to improve segmentation performance.

Local-global concatenation: Zhao et al. [112] proposed a permutation-invariant PS^2-Net to incorporate local structures and global context from point clouds. Edgeconv [87] and NetVLAD [209] are repeatedly stacked to capture the local information and scene-level global features.

Point Convolution Methods. These methods tend to propose effective convolution operators for point clouds. Hua et al. [76] proposed a point-wise convolution operator, where the neighboring points are binned into kernel cells and then convolved with kernel weights. As shown in Fig. 12(b), Wang et al. [201] proposed a network called PCCN based on parametric continuous convolution layers. The kernel function of this layer is parameterized by MLPs and spans the continuous vector space. Thomas et al. [65] proposed a Kernel Point Fully Convolutional Network (KP-FCNN) based on Kernel Point Convolution (KPConv). Specifically, the convolution weights of KPConv are determined by the Euclidean distances to kernel points, and the number of kernel points is not fixed. The positions of the kernel points are formulated as an optimization problem of best coverage in a sphere space. Note that, the radius neighbourhood is used to keep a consistent receptive field, while grid subsampling is used in each layer to achieve high robustness under varying densities of point clouds. In [211], Engelmann et al. provided rich ablation experiments and visualization results to show the impact of receptive field on the performance of aggregation-based methods. They also proposed a Dilated Point Convolution (DPC) operation to aggregate dilated neighboring features, instead of the K nearest neighbours. This operation is demonstrated to be very effective in increasing the receptive field and can be easily integrated into existing aggregation-based networks.

RNN-based Methods. To capture inherent context features from point clouds, Recurrent Neural Networks (RNN) have also been used for semantic segmentation of point clouds. Based on PointNet [5], Engelmann et al. [213] first transformed a block of points into multi-scale blocks and grid blocks to obtain input-level context. Then, the block-wise features extracted by PointNet are sequentially fed into Consolidation Units (CU) or Recurrent Consolidation Units (RCU) to obtain output-level context. Experimental results show that incorporating spatial context is important for the improvement of the segmentation performance. Huang et al. [212] proposed a lightweight local dependency modeling module, and utilized a slice pooling layer to convert unordered point feature sets into an ordered sequence of feature vectors. As shown in Fig. 12(c), Ye et al. [202] first proposed a Pointwise Pyramid Pooling (3P) module to capture the coarse-to-fine local structure, and then utilized two-direction hierarchical RNNs to further obtain long-range spatial dependencies. RNN is then applied to achieve an end-to-end learning. However, these methods lose rich geometric features and density distribution from point clouds when aggregating the local neighbourhood features with global structure features [220]. To alleviate the problems caused by the rigid and static pooling operations, Zhao et al. [220] proposed a Dynamic Aggregation Network (DAR-Net) to consider both global scene complexity and local geometric features. The inter-medium features are dynamically aggregated using a self-adapted receptive field and node weights. Liu et al. [221] proposed 3DCNN-DQN-RNN for efficient semantic parsing of large-scale point clouds. This network first learns the spatial distribution and color features using a 3D CNN network, DQN is further used to localize objects belonging to a specific class. The final concatenated feature vector is fed into a residual RNN to obtain the final segmentation results.

Graph-based Methods. To capture the underlying shapes and geometric structures of 3D point clouds, several methods resort to graph networks. As shown in Fig. 12(d), Landrieu et al. [203] represented a point cloud as a set of interconnected simple shapes and superpoints, and used an attributed directed graph (i.e., superpoint graph) to capture the structure and context information. Then, the large-scale point cloud segmentation problem is spilt into three subproblems, i.e., geometrically homogeneous partition, superpoint embedding, and contextual segmentation. To further improve the partition step, Landrieu and Boussaha [214] proposed a supervised framework to oversegment a point cloud into pure superpoints. This problem is formulated as a deep metric learning problem structured by an adjacency graph. In addition, a graph-structured contrastive loss is also proposed to help the recognition of borders between objects.

To better capture the local geometric relationships in high-dimensional space, Kang et al. [222] proposed a PyramNet based on Graph Embedding Module (GEM) and Pyramid Attention Network (PAN). The GEM module formulates a point cloud as a directed acyclic graph and utilzes a covariance matrix to replace the Euclidean distance for the construction of adjacent similarity matrix. Convolution kernels with four different sizes are used in the PAN module to extract features with different semantic intensities. In [215], Graph Attention Convolution (GAC) is proposed to selectively learn relevant features from a local neighboring set. This operation is achieved by dynamically assigning attention weights to different neighboring points and feature channels based on their spatial positions and feature differences. GAC can learn to capture discriminative features for segmentation, and has similar characteristics to the commonly used CRF model. Ma et al. [223] proposed a Point Global Context Reasoning (PointGCR) module to capture global contextual information along the channel dimension

TABLE 5: Comparative semantic segmentation results on the S3DIS (including both Area5 and 6-fold cross validation) [10], Semantic3D (including both *semantic-8* and *reduced-8* subsets) [12] , ScanNet [11], and SemanticKITTI [15] datasets. Overall Accuracy (OA), Mean Intersection over Union (mIoU) are the main evaluation metric. For simplicity, we omit the '%' after the value. The symbol '-' means the results are unavailable.

Method			**S3DIS**				**Semantic3D**				**ScanNet(v2)**		**Sem. KITTI**
			Area5 (OA)	**Area5 (mIoU)**	**6-fold (mIoU)**	**6-fold (mIoU)**	**sem. (OA)**	**sem. (mIoU)**	**red. (OA)**	**red. (mIoU)**	**OA**	**mIoU**	**(mIoU)**
Projection -based Methods	Multi-view	DeePr3SS [181]	-	-	-	-	-	-	88.9	58.5	-	-	-
		SnapNet [182]	-	-	-	-	91.0	67.4	88.6	59.1	-	-	-
		TangentConv [193]	82.5	52.8	-	-	-	-	-	-	80.1	40.9	40.9
	Spherical	SqueezeSeg [183]	-	-	-	-	-	-	-	-	-	-	29.5
		SqueezeSegV2 [184]	-	-	-	-	-	-	-	-	-	-	39.7
		RangeNet++ [185]	-	-	-	-	-	-	-	-	-	-	52.2
Discretization -based Methods	Volumetric	SEGCloud [196]	-	48.9	-	-	-	-	88.1	61.3	-	-	-
		SparseConvNet [166]	-	-	-	-	-	-	-	-	-	72.5	-
		MinkowskiNet [199]	-	-	-	-	-	-	-	-	-	73.6	-
		VV-Net [186]	-	-	87.8	78.2	-	-	-	-	-	-	-
	Permutohedral lattice	SPLATNet [188]	-	-	-	-	-	-	-	-	-	39.3	18.4
		LatticeNet [189]	-	-	-	-	-	-	-	-	-	64.0	52.2
Hybrid Methods	Hybrid	3DMV [190]	-	-	-	-	-	-	-	-	-	48.4	-
		UPB [200]	-	-	-	-	-	-	-	-	-	63.4	-
		MVPNet [191]	-	-	-	-	-		-	-	-	64.1	-
Point -based Methods	Point-wise MLP	PointNet [5]	-	41.1	78.6	47.6	-	-	-	-	-	-	14.6
		PointNet++ [54]	-	-	81.0	54.5	85.7	63.1	-	-	84.5	33.9	20.1
		PointSIFT [141]	-	-	88.7	70.2	-	-	-	-	86.2	41.5	-
		Engelmann [210]	84.2	52.2	84.0	58.3	-	-	-	-	-	-	-
		3DContextNet [107]	-	-	84.9	55.6	-	-	-	-	-	-	-
		A-SCN [109]	-	-	81.6	52.7	-	-	-	-	-	-	-
		PointWeb [57]	87.0	60.3	87.3	66.7	-	-	-	-	85.9	-	-
		PAT [56]		60.1		64.3	-	-	-	-	-	-	-
		LSANet [207]	-	-	86.8	62.2	-	-	-	-	85.1	-	-
		ShellNet [205]	-	-	87.1	66.8	-	-	93.2	69.3	85.2	-	-
		RandLA-Net [206]	-	-	88.0	70.0	94.6	74.8	94.8	77.4	-	-	55.9
	Point convolution	PointCNN [79]	85.9	57.3	88.1	65.4	-	-	-	-	85.1	45.8	-
		PCCN [201]	-	58.3	-	-	-	-	-	-	-	-	-
		A-CNN [82]	-	-	87.3	-	-	-	-	-	85.4	-	-
		ConvPoint [66]	-	-	88.8	68.2	93.4	76.5	-	-	-	-	-
		KPConv [65]	-	67.1	-	70.6	-	-	92.9	74.6	-	68.4	-
		DPC [211]	86.8	61.3	-	-	-	-	-	-	-	59.2	-
		InterpCNN [80]	-	-	88.7	66.7	-	-	-	-	-	-	-
	RNN -based	RSNet [212]	-	51.9	-	56.5	-	-	-	-	84.9	39.4	-
		G+RCU [213]	-	45.1	81.1	49.7	-	-	-	-	-	-	-
		3P-RNN [202]	85.7	53.4	86.9	56.3	-	-	-	-	-	-	-
	Graph -based	DGCNN [87]	-	-	84.1	56.1	-	-	-	-	-	-	-
		SPG [203]	86.4	58.0	85.5	62.1	92.9	76.2	94.0	73.2	-	-	17.4
		SSP+SPG [214]	87.9	61.7	87.9	68.4	-	-	-	-	-	-	-
		GACNet [215]	87.8	62.9	-	-	-	-	91.9	70.8	-	-	-
		PAG [216]	86.8	59.3	88.1	65.9	-	-	-	-	-	-	
		HDGCN [217]	-	59.3	-	66.9	-	-	-	-	-	-	-
		HPEIN [218]	87.2	61.9	88.2	67.8	-	-	-	-	-	61.8	-
		SPH3D-GCN [219]	87.7	59.5	88.6	68.9	-	-	-	-	-	61.0	-
		DPAM [92]	86.1	60.0	87.6	64.5	-	-	-	-	-	-	-

using an undirected graph representation. PointGCR is a plug-and-play and end-to-end trainable module. It can easily be integrated into an existing segmentation network to achieve performance improvement.

In addition, several very recent work tries to achieve semantic segmentation of point clouds under weak supervision. Wei et al. [224] proposed a two-stage approach to train a segmentation network with subcloud level labels. Xu et al. [225] investigated several inexact supervision schemes for semantic segmentation of point clouds. They also proposed a network that is able to be trained with only partially labeled points (e.g. 10%).

5.2 Instance Segmentation

Compared to semantic segmentation, instance segmentation is more challenging as it requires more accurate and fine-grained reasoning of points. In particular, it not only needs to distinguish the points with different semantic meanings, but also separate instances with the same semantic meaning. Overall, existing methods can be divided into two groups: proposal-based methods and proposal-free methods. Several milestone methods are illustrated in Fig 13.

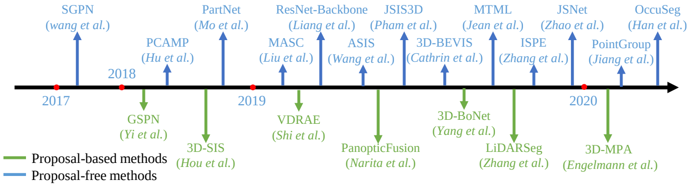


Fig. 13: Chronological overview of the most relevant deep learning-based 3D instance segmentation methods.

5.2.1 Proposal-based Methods

These methods convert the instance segmentation problem into two sub-tasks: 3D object detection and instance mask prediction.

Hou et al. [226] proposed a 3D fully-convolutional Semantic Instance Segmentation (3D-SIS) network to achieve semantic instance segmentation on RGB-D scans. This network learns from both color and geometry features. Similar to 3D object detection, a 3D Region Proposal Network (3D-RPN) and a 3D Region of Interesting (3D-RoI) layer are used to predict bounding box locations, object class labels and instance masks. Following the analysis-by-synthesis strategy, Yi et al. [227] proposed a Generative Shape Proposal Network (GSPN) to generate high-objectness 3D proposals. These proposals are further refined by a Region-based PointNet (R-PointNet). The final label is obtained by predicting a per-point binary mask for each class label. Different from direct regression of 3D bounding boxes from point clouds, this method removes a large amount of meaningless proposals by enforcing geometric understanding.

By extending 2D panoptic segmentation to 3D mapping, Narita et al. [228] proposed an online volumetric 3D mapping system to jointly achieve large-scale 3D reconstruction, semantic labeling, and instance segmentation. They first utilized 2D semantic and instance segmentation networks to obtain pixel-wise panoptic labels and then integrated these labels to the volumtric map. A fully-connected CRF is further used to achieve accurate segmentation. This semantic mapping system can achieve high-quality semantic mapping and discriminative object recognition. Yang et al. [229] proposed a single-stage, anchor-free and end-to-end trainable network called 3D-BoNet to achieve instance segmentation on point clouds. This method directly regresses rough 3D bounding boxes for all potential instances, and then utilizes a point-level binary classifier to obtain instance labels. Particularly, the bounding box generation task is formulated as an optimal assignment problem. In addition, a multi-criteria loss function is also proposed to regularize the generated bounding boxes. This method does not need any post-processing and is computationally efficient. Zhang et al. [230] proposed a network for instance segmentation of large-scale outdoor LiDAR point clouds. This method learns a feature representation on the bird's-eye view of point clouds using self-attention blocks. The final instance labels are obtained based on the predicted horizontal center and the height limits. Shi et al. [231] proposed a hierarchy-aware Variational Denoising Recursive AutoEncoder (VDRAE) to predict the layout of indoor 3D space. The object proposals are iteratively generated and refined by recursive context aggregation and propagation.

Overall, proposal-based methods [226], [227], [229], [232] are intuitive and straightforward, and the instance segmentation results usually have good objectness. However, these methods require multi-stage training and pruning of redundant proposals. Therefore, they are usually time-consuming and computationally expensive.

5.2.2 Proposal-free Methods

Proposal-free methods [233], [234], [235], [236], [237], [238], [239], [240] do not have an object detection module. Instead, they usually consider instance segmentation as a subsequent clustering step after semantic segmentation. In particular, most existing methods are based on the assumption that points belonging to the same instance should have very similar features. Therefore, these methods mainly focus on discriminative feature learning and point grouping.

In a pioneering work, Wang et al. [233] first introduced a Similarity Group Proposal Network (SGPN). This method first learns a feature and semantic map for each point, and then introduces a similarity matrix to represent the similarity between each paired features. To learn more discriminative features, they use a double-hinge loss to mutually adjust the similarity matrix and semantic segmentation results. Finally, a heuristic and non-maximal suppression method is adopted to merge similar points into instances. Since the construction of a similarity matrix requires large memory consumption, the scalability of this method is limited. Similarly, Liu et al. [237] first leveraged submanifold sparse convolution [166] to predict semantic scores of each voxel and affinity between neighboring voxels. They then introduced a clustering algorithm to group points into instances based on the predicted affinity and the mesh topology. Mo et al. [241] introduced a detection-by-segmentation network in PartNet to achieve instance segmentation. PointNet++ is used as the backbone to predict semantic labels of each point and disjoint instance masks. Further, Liang et al. [238] proposed a structure-aware loss for the learning of discriminative embeddings. This loss considers both the similarity of features and the geometric relations among points. An attention-based graph CNN is further used to adaptively refine the learned features by aggregating different information from neighbors.

Since the semantic category and instance label of a point are usually dependent on each other, several methods have been proposed to couple these two tasks into a single task. Wang et al. [234] integrated these two tasks by introducing an end-to-end and learnable Associatively Segmenting Instances and Semantics (ASIS) module. Experiments show that semantic features and instance features can mutually support each other to achieve an improved performance through this ASIS module. Similarly, Zhao et al. [242] proposed JSNet to achieve both semantic and instance segmentation. Further, Pham et al. [235] first introduced a Multi-Task Point-wise Network (MT-PNet) to assign a label to each point and regularized the embeddings in the feature space by introducing a discriminative loss [243]. They then fused the predicted semantic labels and embeddings to a Multi-Value Conditional Random Field (MV-CRF) model for joint optimization. Finally, mean-field variational inference is used to produce semantic labels and instance labels. Hu et al. [244] first proposed a Dynamic Region Growing (DRG) method to dynamically separate a point cloud into a set of disjoint patches, and then used an unsupervised K-means++ algorithm to group all these patches. Multi-scale patch segmentation is then performed with the guidance of contextual information between patches. Finally, these labeled patches are merged into object level to obtain final semantic and instance labels.

To achieve instance segmentation on full 3D scenes, Elich et al. [236] presented a hybrid 2D-3D network to jointly learn global consistent instance features from a BEV representation and local geometric features of point clouds. The learned features are then combined to achieve semantic and instance segmentation. Note that, rather than heuristic *GroupMerging* algorithms [233], a more flexible Meanshift [245] algorithm is used to group these points into instances. Alternatively, multi-task learning is also introduced for instance segmentation. Lahoud et al. [246] learned both the unique feature embedding of each instance and the directional information to estimate the object's center. Feature embedding loss and directional loss are proposed to adjust the learned feature embeddings in latent feature space. Mean-shift clustering and non-maximum suppression are adopted to group voxels into instances. This method achieves the state-of-the-art performance on the ScanNet [11] benchmark. Besides, the predicted directional information is particularly useful to determine the boundary of instances. Zhang et al. [247] introduced probabilistic embeddings to instance segmentation of point clouds. This method also incorporates uncertainty estimation and proposes a new loss function for the clustering step. Jiang et al. [240] proposed a PointGroup network, which is composed of a semantic segmentation branch and an offset prediction branch. A dual-set clustering algorithm and the ScoreNet is further utilized to achieve better grouping results.

In summary, proposal-free methods do not require computationally expensive region-proposal components. However, the objectness of instance segments grouped by these methods is usually low since these methods do not explicitly detect object boundaries.

5.3 Part Segmentation

The difficulties for part segmentation of 3D shapes are twofold. First, shape parts with the same semantic label have a large geometric variation and ambiguity. Second, the number of parts in objects with the same semantic meanings may be different.

VoxSegNet [248] is proposed to achieve fine-grained part segmentation on 3D voxelized data under a limited solution. A Spatial Dense Extraction (SDE) module (which consists of stacked atrous residual blocks) is proposed to extract multiscale discriminative features from sparse volumetric data. The learned features are further re-weighted and fused by progressively applying an Attention Feature Aggregation (AFA) module. Kalogerakis et al. [249] combined FCNs and surface-based CRFs to achieve end-to-end 3D part segmentation. They first generated images from multiple views to achieve optimal surface coverage and fed these images into a 2D network to produce confidence maps. Then, these confidence maps are aggregated by a surface-based CRF, which is responsible for a consistent labeling of the entire scene. Yi et al. [250] introduced a Synchronized Spectral CNN (SyncSpecCNN) to perform convolution on irregular and non-isomorphic shape graphs. A spectral parameterization of dilated convolutional kernels and a spectral transformer network is introduced to solve the problem of multi-scale analysis in parts and information sharing across shapes.

Wang et al. [251] first performed shape segmentation on 3D meshes by introducing Shape Fully Convolutional Networks (SFCN) and taking three low-level geometric features as its input. They then utilized voting-based multilabel graph cuts to further refine the segmentation results. Zhu et al. [252] proposed a weakly-supervised CoSegNet for 3D shape co-segmentation. This network takes a collection of unsegmented 3D point cloud shapes as input, and produces shape part labels by iteratively minimizing a group consistency loss. Similar to CRF, a pre-trained partrefinement network is proposed to further refine and denoise part proposals. Chen et al. [253] proposed a Branched AutoEncoder network (BAE-NET) for unsupervised, oneshot and weakly supervised 3D shape co-segmentation. This method formulates the shape co-segmentation task as a representation learning problem and aims at finding the simplest part representations by minimizing the shape reconstruction loss. Based on the encoder-decoder architecture, each branch of this network can learn a compact representation for a specific part shape. The features learned from each branch and the point coordinate are then fed to the decoder to produce a binary value (which indicates whether the point belongs to this part). This method has good generalization ability and can process large 3D shape collections (up to 5000+ shapes). However, it is sensitive to initial parameters and does not incorporate shape semantics into the network, which hinders this method to obtain a robust and stable estimation in each iteration. Yu et al. [254] proposed a top-down recursive part decomposition network (PartNet) for hierarchical shape segmentation. Different from existing methods that segment a shape to a fixed label set, this network formulates part segmentation as a problem of cascade binary labeling, and decompose the input point cloud to an arbitrary number of parts based on the geometric structure. Luo et al. [255] introduced a learning-based grouping framework for the task of zeroshot 3D part segmentation. To improve the cross-category generalization ability, this method tends to learn a grouping policy that restricts the network to learn part-level features within the part local context.

5.4 Summary

Table 5 shows the results achieved by existing methods on public benchmark, including S3DIS [10], Semantic3D [12], ScanNet [39], and SemanticKITTI [15]. The following issues need to be further investigated:

- Thanks to the regular data representation, both projection-based methods and discretization-based methods can leverage the mature network architecture from their 2D image counterparts. However, the main limitation of projection-based methods lies in the information loss caused by 3D-2D projection, while the main bottleneck for discretization-based methods is the cubically increased computational and memory costs caused by the increase of the resolution. To this end, sparse convolution building upon indexing structures would be a feasible solution and worth further exploration.
- Point-based networks are the most frequently investigated methods. However, point representation naturally does not have explicit neighboring information, most existing point-based methods resort to expensive neighbor searching mechanisms (e.g., KNN [79] or ball query [54]). This inherently limits the efficiency of these methods, the recently proposed point-voxel joint representation [256] would be an interesting direction for further investigation.
- Learning from imbalanced data is still a challenging problem in point cloud segmentation. Although several approaches [65], [203], [205] have achieved a remarkable overall performance, their performance on minority classes is still limited. For example, RandLA-Net [206] achieves an overall IoU of 76.0% on the *reduced-8* subset of Semantic3D, but a very low IOU of 41.1% on the class of *hardscape*.

- The majority of existing approaches [5], [54], [79], [205], [207] work on small point clouds (e.g., 1m×1m with 4096 points). In practice, the point clouds acquired by depth sensors are usually immense and large-scale. Therefore, it is desirable to further investigate the problem of efficient segmentation of large-scale point clouds.
- A handful of works [178], [179], [199] have started to learn spatio-temporal information from dynamic point clouds. It is expected that the spatio-temporal information can help to improve the performance of subsequent tasks such as 3D object recognition, segmentation, and completion.

6 Conclusion

This paper has presented a contemporary survey of the state-of-the-art methods for 3D understanding, including 3D shape classification, 3D object detection and tracking, and 3D scene and object segmentation. A comprehensive taxonomy and performance comparison of these methods have been presented. Merits and demerits of various methods are also covered, with potential research directions being listed.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (No. 61972435, 61602499, 61872379), the Natural Science Foundation of Guangdong Province (2019A1515011271), the Science and Technology Innovation Committee of Shenzhen Municipality (JCYJ20190807152209394), the Australian Research Council (Grants DP150100294 and DP150104251), the China Scholarship Council (CSC) and the Academy of Finland.

References

[1] Z. Liang, Y. Guo, Y. Feng, W. Chen, L. Qiao, L. Zhou, J. Zhang, and H. Liu, "Stereo matching using multi-level cost volume and multi-scale feature constancy," IEEE TPAMI, 2019.

[2] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan, "Rotational projection statistics for 3D local surface description and object recognition," IJCV, 2013.

[3] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, and J. Wan, "3D object recognition in cluttered scenes with local surface features: a survey," IEEE TPAMI, 2014.

[4] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in CVPR, 2017.

[5] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in CVPR, 2017.

[6] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D shapeNets: A deep representation for volumetric shapes," in CVPR, 2015.

[7] M. A. Uy, Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung, "Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data," in ICCV, 2019.

[8] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, and H. Su, "ShapeNet: An information-rich 3D model repository," arXiv preprint arXiv:1512.03012, 2015.

[9] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su, "PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding," in CVPR, 2019.

[10] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3D semantic parsing of large-scale indoor spaces," in CVPR, 2016.

[11] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in CVPR, 2017.

[12] T. Hackel, N. Savinov, L. Ladicky, J. Wegner, K. Schindler, and M. Pollefeys, "Semantic3D.net: A new large-scale point cloud classification benchmark," ISPRS, 2017.

[13] X. Song, P. Wang, D. Zhou, R. Zhu, C. Guan, Y. Dai, H. Su, H. Li, and R. Yang, "Apollocar3D: A large 3D car instance understanding benchmark for autonomous driving," in CVPR, 2019.

[14] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving," in CVPR, 2012.

[15] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "SemanticKITTI: A dataset for semantic scene understanding of lidar sequences," in ICCV, 2019.

[16] G. Elbaz, T. Avraham, and A. Fischer, "3D point cloud registration for localization using a deep neural network auto-encoder," in CVPR, 2017, pp. 4631–4640.

[17] A. Zeng, K.-T. Yu, S. Song, D. Suo, E. Walker, A. Rodriguez, and J. Xiao, "Multi-view self-supervised deep learning for 6D pose estimation in the amazon picking challenge," in ICRA, 2017, pp. 1386–1383.

[18] X. Han, H. Laga, and M. Bennamoun, "Image-based 3D object reconstruction: State-of-the-art and trends in the deep learning era," IEEE TPAMI, 2019.

[19] A. Ioannidou, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris, "Deep learning advances in computer vision with 3D data: A survey," ACM Computing Surveys, 2017.

[20] E. Ahmed, A. Saint, A. E. R. Shabayek, K. Cherenkova, R. Das, G. Gusev, D. Aouada, and B. Ottersten, "Deep learning advances on different 3D data representations: A survey," arXiv preprint arXiv:1808.01462, 2018.

[21] Y. Xie, J. Tian, and X. Zhu, "A review of point cloud semantic segmentation," IEEE GRSM, 2020.

[22] M. M. Rahman, Y. Tan, J. Xue, and K. Lu, "Recent advances in 3D object detection in the era of deep neural networks: A survey," IEEE TIP, 2019.

[23] K. Siddiqi, J. Zhang, D. Macrini, A. Shokoufandeh, S. Bouix, and S. Dickinson, "Retrieving articulated 3-D models using medial surfaces," Machine Vision and Applications, vol. 19, no. 4, pp. 261–275, 2008.

[24] M. De Deuge, B. Douillard, C. Hung, and A. Quadros, "Unsupervised feature learning for classification of outdoor 3D scans," in ACRA, 2013.

[25] S. Song, S. P. Lichtenberg, and J. Xiao, "Sun RGB-D: A RGB-D scene understanding benchmark suite," in CVPR, 2015.

[26] A. Patil, S. Malla, H. Gang, and Y.-T. Chen, "The H3D dataset for full-surround 3D multi-object detection and tracking in crowded urban scenes," in ICRA, 2019.

[27] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan et al., "Argoverse: 3D tracking and forecasting with rich maps," in CVPR, 2019.

[28] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska et al., "Lyft level 5 av dataset 2019," 2019.

[29] Q.-H. Pham, P. Sevestre, R. S. Pahwa, H. Zhan, C. H. Pang, Y. Chen, A. Mustafa, V. Chandrasekhar, and J. Lin, "A*3D dataset: Towards autonomous driving in challenging environments," ICRA, 2020.

[30] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, "Scalability in perception for autonomous driving: Waymo open dataset," in CVPR, 2020.

[31] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in CVPR, 2020.

[32] D. Munoz, J. A. Bagnell, N. Vandapel, and M. Hebert, "Contextual classification with functional max-margin markov networks," in CVPR, 2009, pp. 975–982.

[33] F. Rottensteiner, G. Sohn, J. Jung, M. Gerke, C. Baillard, S. Benitez, and U. Breitkopf, "The isprs benchmark on urban object classification and 3D building reconstruction," ISPRS, 2012.

[34] A. Serna, B. Marcotegui, F. Goulette, and J.-E. Deschaud, "Paris-rue-madame database: a 3D mobile laser scanner dataset for

benchmarking urban detection, segmentation and classification methods," in ICRA, 2014.

[35] B. Vallet, M. Brédif, A. Serna, B. Marcotegui, and N. Paparoditis, "Terramobilita/iqmulus urban point cloud analysis benchmark," Computers & Graphics, vol. 49, pp. 126–133, 2015.

[36] X. Roynard, J.-E. Deschaud, and F. Goulette, "Paris-lille-3d: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification," IJRR, 2018.

[37] W. Tan, N. Qin, L. Ma, Y. Li, J. Du, G. Cai, K. Yang, and J. Li, "Toronto-3D: A large-scale mobile lidar dataset for semantic segmentation of urban roadways," arXiv preprint arXiv:2003.08284, 2020.

[38] N. Varney, V. K. Asari, and Q. Graehling, "Dales: A large-scale aerial lidar data set for semantic segmentation," arXiv preprint arXiv:2004.11985, 2020.

[39] H. Lu, X. Chen, G. Zhang, Q. Zhou, Y. Ma, and Y. Zhao, "SCANet: Spatial-channel attention network for 3D object detection," in ICASSP, 2019.

[40] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3D shape recognition," in ICCV, 2015.

[41] T. Yu, J. Meng, and J. Yuan, "Multi-view harmonized bilinear network for 3D object recognition," in CVPR, 2018.

[42] Z. Yang and L. Wang, "Learning relationships for multi-view 3D object recognition," in ICCV, 2019.

[43] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view CNNs for object classification on 3D data," in CVPR, 2016.

[44] Y. Feng, Z. Zhang, X. Zhao, R. Ji, and Y. Gao, "GVCNN: Group-view convolutional neural networks for 3D shape recognition," in CVPR, 2018.

[45] C. Wang, M. Pelillo, and K. Siddiqi, "Dominant set clustering and pooling for multi-view 3D object recognition," BMVC, 2017.

[46] C. Ma, Y. Guo, J. Yang, and W. An, "Learning multi-view representation with LSTM for 3D shape recognition and retrieval," IEEE TMM, 2018.

[47] X. Wei, R. Yu, and J. Sun, "View-gcn: View-based graph convolutional network for 3D shape analysis," in CVPR, 2020.

[48] D. Maturana and S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition," in IROS, 2015.

[49] G. Riegler, A. Osman Ulusoy, and A. Geiger, "OctNet: Learning deep 3D representations at high resolutions," in CVPR, 2017.

[50] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, "O-CNN: Octree-based convolutional neural networks for 3D shape analysis," ACM TOG, 2017.

[51] T. Le and Y. Duan, "PointGrid: A deep network for 3D shape understanding," in CVPR, 2018.

[52] Y. Ben-Shabat, M. Lindenbaum, and A. Fischer, "3D point cloud classification and segmentation using 3D modified fisher vector representation for convolutional neural networks," arXiv preprint arXiv:1711.08241, 2017.

[53] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," in NeurIPS, 2017.

[54] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in NeurIPS, 2017.

[55] M. Joseph-Rivlin, A. Zvirin, and R. Kimmel, "Mo-Net: Flavor the moments in learning to classify shapes," in ICCVW, 2018.

[56] J. Yang, Q. Zhang, B. Ni, L. Li, J. Liu, M. Zhou, and Q. Tian, "Modeling point clouds with self-attention and gumbel subset sampling," in CVPR, 2019.

[57] H. Zhao, L. Jiang, C.-W. Fu, and J. Jia, "PointWeb: Enhancing local neighborhood features for point cloud processing," in CVPR, 2019.

[58] Y. Duan, Y. Zheng, J. Lu, J. Zhou, and Q. Tian, "Structural relational reasoning of point clouds," in CVPR, 2019.

[59] H. Lin, Z. Xiao, Y. Tan, H. Chao, and S. Ding, "Justlookup: One millisecond deep feature extraction for point clouds by lookup tables," in ICME, 2019.

[60] X. Sun, Z. Lian, and J. Xiao, "SRINet: Learning strictly rotation-invariant representations for point cloud classification and segmentation," in ACM MM, 2019.

[61] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, "Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling," in CVPR, 2020.

[62] Y. Liu, B. Fan, S. Xiang, and C. Pan, "Relation-shape convolutional neural network for point cloud analysis," in CVPR, 2019.

[63] A. Boulch, "Generalizing discrete convolutions for unstructured point clouds," arXiv preprint arXiv:1904.02375, 2019.

[64] Y. Liu, B. Fan, G. Meng, J. Lu, S. Xiang, and C. Pan, "DensePoint: Learning densely contextual representation for efficient point cloud processing," in ICCV, 2019.

[65] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in ICCV, 2019.

[66] A. Boulch, "ConvPoint: continuous convolutions for point cloud processing," Computers & Graphics, 2020.

[67] W. Wu, Z. Qi, and L. Fuxin, "PointConv: Deep convolutional networks on 3D point clouds," in CVPR, 2019.

[68] P. Hermosilla, T. Ritschel, P.-P. Vázquez, À. Vinacua, and T. Ropinski, "Monte carlo convolution for learning on non-uniformly sampled point clouds," ACM TOG, 2018.

[69] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "SpiderCNN: Deep learning on point sets with parameterized convolutional filters," in ECCV, 2018.

[70] A. Matan, M. Haggai, and L. Yaron, "Point convolutional neural networks by extension operators," ACM TOG, 2018.

[71] C. Esteves, C. Allen-Blanchette, A. Makadia, and K. Daniilidis, "Learning so(3) equivariant representations with spherical CNNs," in ECCV, 2017.

[72] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley, "Tensor field networks: Rotation-and translation-equivariant neural networks for 3D point clouds," arXiv preprint arXiv:1802.08219, 2018.

[73] T. S. Cohen, M. Geiger, J. Koehler, and M. Welling, "Spherical CNNs," ICLR, 2018.

[74] A. Poulenard, M.-J. Rakotosaona, Y. Ponty, and M. Ovsjanikov, "Effective rotation-invariant point CNN with spherical harmonics kernels," in 3DV, 2019.

[75] F. Groh, P. Wieschollek, and H. P. Lensch, "Flex-Convolution," in ACCV, 2018.

[76] B.-S. Hua, M.-K. Tran, and S.-K. Yeung, "Pointwise convolutional neural networks," in CVPR, 2018.

[77] H. Lei, N. Akhtar, and A. Mian, "Octree guided cnn with spherical kernels for 3D point clouds," in CVPR, 2019.

[78] S. Lan, R. Yu, G. Yu, and L. S. Davis, "Modeling local geometric structure of 3D point clouds using geo-cnn," in CVPR, 2019.

[79] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on x-transformed points," in NeurIPS, 2018.

[80] J. Mao, X. Wang, and H. Li, "Interpolated convolutional networks for 3D point cloud understanding," in ICCV, 2019.

[81] Z. Zhang, B.-S. Hua, D. W. Rosen, and S.-K. Yeung, "Rotation invariant convolutions for 3D point clouds deep learning," in 3DV, 2019.

[82] A. Komarichev, Z. Zhong, and J. Hua, "A-CNN: Annularly convolutional neural networks on point clouds," in CVPR, 2019.

[83] S. Kumawat and S. Raman, "LP-3DCNN: Unveiling local phase in 3D convolutional neural networks," in CVPR, 2019.

[84] Y. Rao, J. Lu, and J. Zhou, "Spherical fractal convolutional neural networks for point cloud recognition," in CVPR, 2019.

[85] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in CVPR, 2017.

[86] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in ICRA, 2011.

[87] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," ACM TOG, 2019.

[88] K. Zhang, M. Hao, J. Wang, C. W. de Silva, and C. Fu, "Linked dynamic graph CNN: Learning on point cloud via linking hierarchical features," arXiv preprint arXiv:1904.10014, 2019.

[89] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point cloud auto-encoder via deep grid deformation," in CVPR, 2018.

[90] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in CVPR, 2015.

[91] K. Hassani and M. Haley, "Unsupervised multi-task feature learning on point clouds," in ICCV, 2019.

[92] J. Liu, B. Ni, C. Li, J. Yang, and Q. Tian, "Dynamic points agglomeration for hierarchical point sets learning," in ICCV, 2019.

[93] Y. Shen, C. Feng, Y. Yang, and D. Tian, "Mining point cloud local structures by kernel correlation and graph pooling," in CVPR, 2018.

[94] M. Dominguez, R. Dhamdhere, A. Petkar, S. Jain, S. Sah, and R. Ptucha, "General-purpose deep point cloud feature extractor," in WACV, 2018.

[95] C. Chen, G. Li, R. Xu, T. Chen, M. Wang, and L. Lin, "ClusterNet: Deep hierarchical cluster network with rigorously rotation-invariant representation for point cloud analysis," in CVPR, 2019.

[96] D. Müllner, "Modern hierarchical, agglomerative clustering algorithms," arXiv preprint arXiv:1109.2378, 2011.

[97] Q. Xu, X. Sun, C.-Y. Wu, P. Wang, and U. Neumann, "Grid-gcn for fast and scalable point cloud learning," in CVPR, 2020.

[98] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, "Spectral networks and locally connected networks on graphs," ICLR, 2014.

[99] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in NeurIPS, 2016.

[100] G. Te, W. Hu, A. Zheng, and Z. Guo, "RGCNN: Regularized graph CNN for point cloud segmentation," in ACM MM, 2018.

[101] R. Li, S. Wang, F. Zhu, and J. Huang, "Adaptive graph convolutional neural networks," in AAAI, 2018.

[102] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in AAAI, 2019.

[103] C. Wang, B. Samari, and K. Siddiqi, "Local spectral graph convolution for point set feature learning," in ECCV, 2018.

[104] Y. Zhang and M. Rabbat, "A Graph-CNN for 3D point cloud classification," in ICASSP, 2018.

[105] G. Pan, J. Wang, R. Ying, and P. Liu, "3DTI-Net: Learn inner transform invariant 3D geometry features using dynamic GCN," arXiv preprint arXiv:1812.06254, 2018.

[106] R. Klokov and V. Lempitsky, "Escape from cells: Deep kd-networks for the recognition of 3D point cloud models," in ICCV, 2017.

[107] W. Zeng and T. Gevers, "3DContextNet: K-d tree guided hierarchical learning of point clouds using local and global contextual cues," in ECCV, 2018.

[108] J. Li, B. M. Chen, and G. Hee Lee, "SO-Net: Self-organizing network for point cloud analysis," in CVPR, 2018.

[109] S. Xie, S. Liu, Z. Chen, and Z. Tu, "Attentional ShapeContextNet for point cloud recognition," in CVPR, 2018.

[110] H. You, Y. Feng, R. Ji, and Y. Gao, "PVNet: A joint convolutional network of point cloud and multi-view for 3D shape recognition," in ACM MM, 2018.

[111] H. You, Y. Feng, X. Zhao, C. Zou, R. Ji, and Y. Gao, "PVRNet: Point-view relation neural network for 3D shape recognition," in AAAI, 2019.

[112] Y. Zhao, T. Birdal, H. Deng, and F. Tombari, "3D point capsule networks," in CVPR, 2019.

[113] W. Chen, X. Han, G. Li, C. Chen, J. Xing, Y. Zhao, and H. Li, "Deep RBFNet: Point cloud feature learning using radial basis functions," arXiv preprint arXiv:1812.04302, 2018.

[114] X. Liu, Z. Han, Y.-S. Liu, and M. Zwicker, "Point2Sequence: Learning the shape representation of 3D point clouds with an attention-based sequence to sequence network," in AAAI, 2019.

[115] P. Wu, C. Chen, J. Yi, and D. Metaxas, "Point cloud processing via recurrent set encoding," in AAAI, 2019.

[116] C. Qin, H. You, L. Wang, C.-C. J. Kuo, and Y. Fu, "PointDAN: A multi-scale 3D domain adaption network for point cloud representation," in NIPS, 2019.

[117] B. Sievers and J. Sauder, "Self-supervised deep learning on point clouds by reconstructing space," in NIPS, 2019.

[118] R. Li, X. Li, P.-A. Heng, and C.-W. Fu, "PointAugment: An auto-augmentation framework for point cloud classification," in CVPR, 2020.

[119] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," IEEE TPAMI, 2002.

[120] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in NeurIPS, 2017.

[121] D. Bobkov, S. Chen, R. Jian, Z. Iqbal, and E. Steinbach, "Noise-resistant deep learning for object classification in 3D point clouds using a point pair descriptor," IEEE RAL, 2018.

[122] S. Prokudin, C. Lassner, and J. Romero, "Efficient learning on point clouds with basis point sets," in ICCV, 2019.

[123] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," IJCV, 2020.

[124] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3D object detection in point clouds," ICCV, 2019.

[125] W. Shi and R. Rajkumar, "Point-GNN: Graph neural network for 3D object detection in a point cloud," in CVPR, 2020.

[126] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3D proposal generation and object detection from view aggregation," in IROS, 2018.

[127] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3D object detection," in ECCV, 2018.

[128] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun, "Multi-task multi-sensor fusion for 3D object detection," in CVPR, 2019.

[129] B. Yang, W. Luo, and R. Urtasun, "PIXOR: Real-time 3D object detection from point clouds," in CVPR, 2018.

[130] W. Luo, B. Yang, and R. Urtasun, "Fast and furious: Real time end-to-end 3D detection, tracking and motion forecasting with a single convolutional net," in CVPR, 2018.

[131] Y. Zeng, Y. Hu, S. Liu, J. Ye, Y. Han, X. Li, and N. Sun, "RT3D: Real-time 3D vehicle detection in lidar point cloud for autonomous driving," IEEE RAL, 2018.

[132] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, "IPOD: Intensive point-based object detector for point cloud," arXiv preprint arXiv:1812.05276, 2018.

[133] S. Shi, X. Wang, and H. Li, "PointRCNN: 3D object proposal generation and detection from point cloud," in CVPR, 2019.

[134] Z. Jesus, G. Silvio, and G. Bernard, "PointRGCN: Graph convolution networks for 3D vehicles detection refinement," arXiv preprint arXiv:1911.12236, 2019.

[135] V. Sourabh, L. Alex H., H. Bassam, and B. Oscar, "PointPainting: Sequential fusion for 3D object detection," in CVPR, 2020.

[136] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in CVPR, 2018.

[137] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in CVPR, 2019.

[138] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, "STD: Sparse-to-dense 3D object detector for point cloud," in ICCV, 2019.

[139] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum PointNets for 3D object detection from RGB-D data," in CVPR, 2018.

[140] X. Zhao, Z. Liu, R. Hu, and K. Huang, "3D object detection using scale invariant and feature reweighting networks," in AAAI, 2019.

[141] M. Jiang, Y. Wu, and C. Lu, "PointSIFT: A sift-like network module for 3D point cloud semantic segmentation," arXiv preprint arXiv:1807.00652, 2018.

[142] D. Xu, D. Anguelov, and A. Jain, "PointFusion: Deep sensor fusion for 3D bounding box estimation," in CVPR, 2018.

[143] K. Shin, Y. P. Kwon, and M. Tomizuka, "RoarNet: A robust 3D object detection based on region approximation refinement," in IEEE IV, 2019.

[144] Z. Wang and K. Jia, "Frustum convNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection," in IROS, 2019.

[145] L. Johannes, M. Andreas, A. Thomas, H. Markus, N. Bernhard, and H. Sepp, "Patch refinement - localized 3D object detection," arXiv preprint arXiv:1910.04093, 2019.

[146] D. Zhou, J. Fang, X. Song, C. Guan, J. Yin, Y. Dai, and R. Yang, "Iou loss for 2D/3D object detection," in 3DV, 2019.

[147] Y. Chen, S. Liu, X. Shen, and J. Jia, "Fast point r-cnn," in ICCV, 2019.

[148] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "PV-RCNN: Point-voxel feature set abstraction for 3D object detection," in CVPR, 2020.

[149] M. Feng, S. Z. Gilani, Y. Wang, L. Zhang, and A. Mian, "Relation graph network for 3D object detection in point clouds," arXiv preprint arXiv:1912.00202, 2019.

[150] C. R. Qi, X. Chen, O. Litany, and L. J. Guibas, "ImVoteNet: Boosting 3D object detection in point clouds with image votes," in CVPR, 2020.

[151] S. Shi, Z. Wang, X. Wang, and H. Li, "From points to parts: 3D object detection from point cloud with part-aware and part-aggregation network," TPAMI, 2020.

[152] B. Yang, M. Liang, and R. Urtasun, "HDNET: Exploiting hd maps for 3D object detection," in CoRL, 2018.

[153] J. Beltrán, C. Guindel, F. M. Moreno, D. Cruzado, F. García, and A. De La Escalera, "BirdNet: a 3D object detection framework from lidar information," in ITSC, 2018.

[154] B. Li, T. Zhang, and T. Xia, "Vehicle detection from 3D lidar using fully convolutional network," arXiv preprint arXiv:1608.07916, 2016.

[155] B. Li, "3D fully convolutional network for vehicle detection in point cloud," in IROS, 2017.

[156] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, "Vote3Deep: Fast object detection in 3D point clouds using efficient convolutional neural networks," in ICRA, 2017.

[157] X. Li, J. E. Guivant, N. Kwok, and Y. Xu, "3D backbone network for 3D object detection," in CoRR, 2019.

[158] Y. Yan, Y. Mao, and B. Li, "SECOND: Sparsely embedded convolutional detection," Sensors, 2018.

[159] V. A. Sindagi, Y. Zhou, and O. Tuzel, "MVX-Net: Multimodal voxelnet for 3D object detection," in ICRA, 2019.

[160] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. Zhang, "Structure aware single-stage 3D object detection from point cloud," in CVPR, 2020.

[161] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3DSSD: Point-based 3D single stage object detector," in CVPR, 2020.

[162] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "LaserNet: An efficient probabilistic 3D object detector for autonomous driving," CVPR, 2019.

[163] G. P. Meyer, J. Charland, D. Hegde, A. Laddha, and C. Vallespi-Gonzalez, "Sensor fusion for joint 3D object detection and semantic segmentation," CVPRW, 2019.

[164] Q. Chen, L. Sun, Z. Wang, K. Jia, and A. Yuille, "Object as hotspots: An anchor-free 3D object detection approach via firing of hotspots," arXiv preprint arXiv:1912.12791, 2019.

[165] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in MICCAI, 2015, pp. 234–241.

[166] B. Graham, M. Engelcke, and L. van der Maaten, "3D semantic segmentation with submanifold sparse convolutional networks," in CVPR, 2018.

[167] Q. Hu, Y. Guo, Y. Chen, J. Xiao, and W. An, "Correlation filter tracking: Beyond an open-loop system," in BMVC, 2017.

[168] H. Liu, Q. Hu, B. Li, and Y. Guo, "Robust long-term tracking via instance specific proposals," IEEE TIM, 2019.

[169] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in ECCV, 2016.

[170] S. Giancola, J. Zarzar, and B. Ghanem, "Leveraging shape completion for 3D siamese tracking," CVPR, 2019.

[171] M. Mueller, N. Smith, and B. Ghanem, "Context-aware correlation filter tracking," in CVPR, 2017.

[172] J. Zarzar, S. Giancola, and B. Ghanem, "Efficient tracking proposals using 2D-3D siamese networks on lidar," arXiv preprint arXiv:1903.10168, 2019.

[173] M. Simon, K. Amende, A. Kraus, J. Honer, T. Sämann, H. Kaulbersch, S. Milz, and H. M. Gross, "Complexer-YOLO: Real-time 3D object detection and tracking on semantic point clouds," CVPRW, 2019.

[174] H. Qi, C. Feng, Z. Cao, F. Zhao, and Y. Xiao, "P2B: Point-to-box network for 3D object tracking in point clouds," in CVPR, 2020.

[175] X. Liu, C. R. Qi, and L. J. Guibas, "FlowNet3D: Learning scene flow in 3D point clouds," in CVPR, 2019.

[176] Z. Wang, S. Li, H. Howard-Jenkins, V. Prisacariu, and M. Chen, "FlowNet3D++: Geometric losses for deep scene flow estimation," in WACV, 2020.

[177] X. Gu, Y. Wang, C. Wu, Y. J. Lee, and P. Wang, "HPLFlowNet: Hierarchical permutohedral lattice flowNet for scene flow estimation on large-scale point clouds," in CVPR, 2019.

[178] H. Fan and Y. Yang, "PointRNN: Point recurrent neural network for moving point cloud processing," arXiv preprint arXiv:1910.08287, 2019.

[179] X. Liu, M. Yan, and J. Bohg, "MeteorNet: Deep learning on dynamic 3D point cloud sequences," in ICCV, 2019.

[180] H. Mittal, B. Okorn, and D. Held, "Just go with the flow: Self-supervised scene flow estimation," in CVPR, 2020.

[181] F. J. Lawin, M. Danelljan, P. Tosteberg, G. Bhat, F. S. Khan, and M. Felsberg, "Deep projective 3D semantic segmentation," in CAIP, 2017.

[182] A. Boulch, B. Le Saux, and N. Audebert, "Unstructured point cloud semantic labeling using deep segmentation networks." in 3DOR, 2017.

[183] B. Wu, A. Wan, X. Yue, and K. Keutzer, "SqueezeSeg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3D lidar point cloud," in ICRA, 2018.

[184] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud," in ICRA, 2019.

[185] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and accurate lidar semantic segmentation," in IROS, 2019.

[186] H.-Y. Meng, L. Gao, Y.-K. Lai, and D. Manocha, "VV-Net: Voxel vae net with group convolutions for point cloud segmentation," in ICCV, 2019.

[187] D. Rethage, J. Wald, J. Sturm, N. Navab, and F. Tombari, "Fully-convolutional point networks for large-scale point clouds," in ECCV, 2018.

[188] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M.-H. Yang, and J. Kautz, "SplatNet: Sparse lattice networks for point cloud processing," in CVPR, 2018.

[189] R. A. Rosu, P. Schütt, J. Quenzel, and S. Behnke, "LatticeNet: Fast point cloud segmentation using permutohedral lattices," arXiv preprint arXiv:1912.05905, 2019.

[190] A. Dai and M. Nießner, "3DMV: Joint 3D-multi-view prediction for 3D semantic scene segmentation," in ECCV, 2018.

[191] M. Jaritz, J. Gu, and H. Su, "Multi-view pointNet for 3D scene understanding," in ICCVW, 2019.

[192] N. Audebert, B. Le Saux, and S. Lefèvre, "Semantic segmentation of earth observation data using multimodal and multi-scale deep networks," in ACCV, 2016.

[193] M. Tatarchenko, J. Park, V. Koltun, and Q.-Y. Zhou, "Tangent convolutions for dense prediction in 3D," in CVPR, 2018.

[194] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and < 0.5 MB model size," in ICLR, 2016.

[195] J. Huang and S. You, "Point cloud labeling using 3D convolutional neural network," in ICPR, 2016.

[196] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "SEGCloud: Semantic segmentation of 3D point clouds," in 3DV, 2017.

[197] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in CVPR, 2015.

[198] A. Dai, D. Ritchie, M. Bokeloh, S. Reed, J. Sturm, and M. Nießner, "ScanComplete: Large-scale scene completion and semantic segmentation for 3D scans," in CVPR, 2018.

[199] C. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal convnets: Minkowski convolutional neural networks," in CVPR, 2019.

[200] H.-Y. Chiang, Y.-L. Lin, Y.-C. Liu, and W. H. Hsu, "A unified point-based framework for 3D segmentation," in 3DV, 2019.

[201] S. Wang, S. Suo, W.-C. Ma, A. Pokrovsky, and R. Urtasun, "Deep parametric continuous convolutional neural networks," in CVPR, 2018.

[202] X. Ye, J. Li, H. Huang, L. Du, and X. Zhang, "3D recurrent neural networks with context fusion for point cloud semantic segmentation," in ECCV, 2018.

[203] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in CVPR, 2018.

[204] F. Engelmann, T. Kontogianni, J. Schult, and B. Leibe, "Know what your neighbors do: 3D semantic segmentation of point clouds," in ECCVW, 2018.

[205] Z. Zhang, B.-S. Hua, and S.-K. Yeung, "ShellNet: Efficient point cloud convolutional neural networks using concentric shells statistics," in ICCV, 2019.

[206] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," CVPR, 2020.

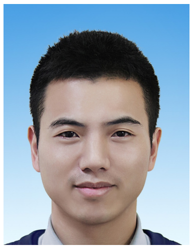
[207] L.-Z. Chen, X.-Y. Li, D.-P. Fan, M.-M. Cheng, K. Wang, and S.-P. Lu, "LSANet: Feature learning on point sets by local spatial attention," arXiv preprint arXiv:1905.05442, 2019.

[208] C. Zhao, W. Zhou, L. Lu, and Q. Zhao, "Pooling scores of neighboring points for improved 3D point cloud segmentation," in ICIP, 2019.

[209] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in CVPR, 2016.

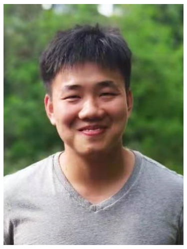
[210] F. Engelmann, T. Kontogianni, J. Schult, and B. Leibe, "Know what your neighbors do: 3D semantic segmentation of point clouds," in ECCV, 2018.

[211] F. Engelmann, T. Kontogianni, and B. Leibe, "Dilated point convolutions: On the receptive field of point convolutions," in ICRA, 2020.
[212] Q. Huang, W. Wang, and U. Neumann, "Recurrent slice networks for 3D segmentation of point clouds," in CVPR, 2018.
[213] F. Engelmann, T. Kontogianni, A. Hermans, and B. Leibe, "Exploring spatial context for 3D semantic segmentation of point clouds," in ICCV, 2017.
[214] L. Landrieu and M. Boussaha, "Point cloud oversegmentation with graph-structured deep metric learning," in CVPR, 2019.
[215] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, "Graph attention convolution for point cloud semantic segmentation," in CVPR, 2019.
[216] L. Pan, C.-M. Chew, and G. H. Lee, "Pointatrousgraph: Deep hierarchical encoder-decoder with atrous convolution for point clouds," arXiv preprint arXiv:1907.09798, 2019.
[217] Z. Liang, M. Yang, L. Deng, C. Wang, and B. Wang, "Hierarchical depthwise graph convolutional neural network for 3D semantic segmentation of point clouds," in ICRA, 2019.
[218] L. Jiang, H. Zhao, S. Liu, X. Shen, C.-W. Fu, and J. Jia, "Hierarchical point-edge interaction network for point cloud semantic segmentation," in ICCV, 2019.
[219] H. Lei, N. Akhtar, and A. Mian, "Spherical convolutional neural network for 3D point clouds," arXiv preprint arXiv:1805.07872, 2018.
[220] Z. Zhao, M. Liu, and K. Ramani, "DAR-Net: Dynamic aggregation network for semantic scene segmentation," arXiv preprint arXiv:1907.12022, 2019.
[221] F. Liu, S. Li, L. Zhang, C. Zhou, R. Ye, Y. Wang, and J. Lu, "3DCNN-DQN-RNN: A deep reinforcement learning framework for semantic parsing of large-scale 3D point clouds," in ICCV, 2017.
[222] Z. Kang and N. Li, "PyramNet: Point cloud pyramid attention network and graph embedding module for classification and segmentation," in ICONIP, 2019.
[223] Y. Ma, Y. Guo, H. Liu, Y. Lei, and G. Wen, "Global context reasoning for semantic segmentation of 3D point clouds," in WACV, 2020.
[224] J. Wei, G. Lin, K.-H. Yap, T.-Y. Hung, and L. Xie, "Multi-path region mining for weakly supervised 3D semantic segmentation on point clouds," in CVPR, 2020.
[225] X. Xu and G. H. Lee, "Weakly supervised semantic point cloud segmentation: Towards 10x fewer labels," in CVPR, 2020, pp. 13 706–13 715.
[226] J. Hou, A. Dai, and M. Nießner, "3D-SIS: 3D semantic instance segmentation of RGB-D scans," in CVPR, 2019.
[227] L. Yi, W. Zhao, H. Wang, M. Sung, and L. J. Guibas, "GSPN: Generative shape proposal network for 3D instance segmentation in point cloud," in CVPR, 2019.
[228] G. Narita, T. Seno, T. Ishikawa, and Y. Kaji, "PanopticFusion: Online volumetric semantic mapping at the level of stuff and things," in IROS, 2019.
[229] B. Yang, J. Wang, R. Clark, Q. Hu, S. Wang, A. Markham, and N. Trigoni, "Learning object bounding boxes for 3D instance segmentation on point clouds," in NeurIPS, 2019.
[230] F. Zhang, C. Guan, J. Fang, S. Bai, R. Yang, P. Torr, and V. Prisacariu, "Instance segmentation of lidar point clouds," in ICRA, 2020.
[231] Y. Shi, A. X. Chang, Z. Wu, M. Savva, and K. Xu, "Hierarchy denoising recursive autoencoders for 3D scene layout prediction," in CVPR, 2019.
[232] F. Engelmann, M. Bokeloh, A. Fathi, B. Leibe, and M. Nießner, "3d-mpa: Multi-proposal aggregation for 3d semantic instance segmentation," in CVPR, 2020.
[233] W. Wang, R. Yu, Q. Huang, and U. Neumann, "SGPN: Similarity group proposal network for 3D point cloud instance segmentation," in CVPR, 2018.
[234] X. Wang, S. Liu, X. Shen, C. Shen, and J. Jia, "Associatively segmenting instances and semantics in point clouds," in CVPR, 2019.
[235] Q.-H. Pham, T. Nguyen, B.-S. Hua, G. Roig, and S.-K. Yeung, "JSIS3D: Joint semantic-instance segmentation of 3D point clouds with multi-task pointwise networks and multi-value conditional random fields," in CVPR, 2019.
[236] C. Elich, F. Engelmann, J. Schult, T. Kontogianni, and B. Leibe, "3D-BEVIS: Birds-eye-view instance segmentation," in GCPR, 2019.
[237] C. Liu and Y. Furukawa, "MASC: Multi-scale affinity with sparse convolution for 3D instance segmentation," arXiv preprint arXiv:1902.04478, 2019.
[238] Z. Liang, M. Yang, and C. Wang, "3D graph embedding learning with a structure-aware loss function for point cloud semantic instance segmentation," arXiv preprint arXiv:1902.05247, 2019.
[239] L. Han, T. Zheng, L. Xu, and L. Fang, "Occuseg: Occupancy-aware 3d instance segmentation," in CVPR, 2020.
[240] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, and J. Jia, "PointGroup: Dual-set point grouping for 3D instance segmentation," in CVPR, 2020.
[241] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su, "PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding," in CVPR, 2019.
[242] L. Zhao and W. Tao, "JSNet: Joint instance and semantic segmentation of 3D point clouds," in AAAI, 2020.
[243] B. De Brabandere, D. Neven, and L. Van Gool, "Semantic instance segmentation with a discriminative loss function," in CVPRW, 2017.
[244] S.-M. Hu, J.-X. Cai, and Y.-K. Lai, "Semantic labeling and instance segmentation of 3D point clouds using patch context analysis and multiscale processing," IEEE TVCG, 2018.
[245] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," IEEE TPAMI, 2002.
[246] J. Lahoud, B. Ghanem, M. Pollefeys, and M. R. Oswald, "3D instance segmentation via multi-task metric learning," in ICCV, 2019.
[247] B. Zhang and P. Wonka, "Point cloud instance segmentation using probabilistic embeddings," arXiv preprint arXiv:1912.00145, 2019.
[248] Z. Wang and F. Lu, "VoxSegNet: Volumetric CNNs for semantic part segmentation of 3D shapes," IEEE TVCG, 2019.
[249] E. Kalogerakis, M. Averkiou, S. Maji, and S. Chaudhuri, "3D shape segmentation with projective convolutional networks," in CVPR, 2017.
[250] L. Yi, H. Su, X. Guo, and L. J. Guibas, "SyncSpecCNN: Synchronized spectral CNN for 3D shape segmentation," in CVPR, 2017.
[251] P. Wang, Y. Gan, P. Shui, F. Yu, Y. Zhang, S. Chen, and Z. Sun, "3D shape segmentation via shape fully convolutional networks," Computers & Graphics, 2018.
[252] C. Zhu, K. Xu, S. Chaudhuri, L. Yi, L. Guibas, and H. Zhang, "CoSegNet: Deep co-segmentation of 3D shapes with group consistency loss," arXiv preprint arXiv:1903.10297, 2019.
[253] Z. Chen, K. Yin, M. Fisher, S. Chaudhuri, and H. Zhang, "BAE-NET: Branched autoencoder for shape co-segmentation," in ICCV, 2019.
[254] F. Yu, K. Liu, Y. Zhang, C. Zhu, and K. Xu, "PartNet: A recursive part decomposition network for fine-grained and hierarchical shape segmentation," in CVPR, 2019.
[255] T. Luo, K. Mo, Z. Huang, J. Xu, S. Hu, L. Wang, and H. Su, "Learning to group: A bottom-up framework for 3D part discovery in unseen categories," in ICLR, 2020.
[256] Z. Liu, H. Tang, Y. Lin, and S. Han, "Point-Voxel CNN for efficient 3D deep learning," in NeurIPS, 2019.



Yulan Guo is currently as associate professor. He received the B.Eng. and Ph.D. degrees from National University of Defense Technology (NUDT) in 2008 and 2015, respectively. He was a visiting Ph.D. student with the University of Western Australia from 2011 to 2014. He worked as a postdoctorial research fellow with the Institute of Computing Technology, Chinese Academy of Sciences from 2016 to 2018. He has authored over 90 articles in journals and conferences, such as the IEEE TPAMI and IJCV. His current research interests focus on 3D vision, particularly on 3D feature learning, 3D modeling, 3D object recognition, and scene understanding. Dr. Guo received the ACM China SIGAI Rising Star Award in 2019, Wu-Wenjun Outstanding AI Youth Award in 2019, and the CAAI Outstanding Doctoral Dissertation Award in 2016. He served as an associate editor for IET Computer Vision and IET Image Processing, a guest editor for IEEE TPAMI, and an area chair for CVPR 2021 and ICPR 2020.

Hanyun Wang received his Ph.D. degree from National University of Defense Technology in 2015. He was a visiting Ph.D. student with Xiamen University from 2011 to 2014. He has authored over 20 articles in journals and conferences, such as IEEE TGRS and IEEE TITS. His research interests include mobile laser scanning data analysis and 3D computer vision, especially on 3D object detection and 3D scene understanding. He also served as reviewers for many journals, such as IEEE TGRS, IEEE GRSL and IET Image Processing.



Qingyong Hu received his M.Eng. degree in information and communication engineering from the National University of Defense Technology (NUDT) in 2018. He is currently a DPhil candidate in the Department of Computer Science at the University of Oxford. His research interests lie in 3D computer vision, large-scale point cloud processing, and visual tracking.

Hao Liu received the B.Eng. degree from University of Electronic Science and Technology of China (UESTC) in 2016, and M.S. degree from National University of Defense Technology (NUDT) in 2018. He is currently pursuing the Ph.D. degree with the School of Electronics and Communication Engineering, Sun Yat-sen University. His research interests lie in 3D computer vision and point cloud processing.

Li Liu received the BSc degree in communication engineering, the MSc degree in photogrammetry and remote sensing and the Ph.D. degree in information and communication engineering from the National University of Defense Technology (NUDT), China, in 2003, 2005 and 2012, respectively. She joined the faculty at NUDT in 2012, where she is currently an Associate Professor with the College of System Engineering. During her PhD study, she spent more than two years as a Visiting Student at the University of Waterloo, Canada, from 2008 to 2010. From 2015 to 2016, she spent ten months visiting the Multimedia Laboratory at the Chinese University of Hong Kong. From 2016.12 to 2018.11, she worked as a senior researcher at the Machine Vision Group at the University of Oulu, Finland. She was a cochair of nine International Workshops at CVPR, ICCV, and ECCV. She was a guest editor of special issues for IEEE TPAMI and IJCV. Her current research interests include computer vision, pattern recognition and machine learning. Her papers have currently over 2300+ citations in Google Scholar. She currently serves as Associate Editor of the Visual Computer Journal and Pattern Recognition Letter. She serves as Area Chair of ICME 2020.

Mohammed Bennamoun is Winthrop Professor in the Department of Computer Science and Software Engineering at UWA and is a researcher in computer vision, machine/deep learning, robotics, and signal/speech processing. He has published 4 books (available on Amazon), 1 edited book, 1 Encyclopedia article, 14 book chapters, 120+ journal papers, 250+ conference publications, 16 invited & keynote publications. His h-index is 50 and his number of citations is 11,000+ (Google Scholar). He was awarded 65+ competitive research grants, from the Australian Research Council, and numerous other Government, UWA and industry Research Grants. He successfully supervised 26+ PhD students to completion. He won the Best Supervisor of the Year Award at QUT (1998), and received award for research supervision at UWA (2008 & 2016) and Vice-Chancellor Award for mentorship (2016). He delivered conference tutorials at major conferences, including: IEEE Computer Vision and Pattern Recognition (CVPR 2016), Interspeech 2014, IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP) and European Conference on Computer Vision (ECCV). He was also invited to give a Tutorial at an International Summer School on Deep Learning (DeepLearn 2017).