Point2Seq: Detecting 3D Objects as Sequences

Yujing Xue[1,2*] Jiageng Mao[3*] Minzhe Niu[2] Hang Xu[2] Michael Bi Mi[4]
Wei Zhang[2] Xiaogang Wang[3] Xinchao Wang[1†]
[1]National University of Singapore [2]Huawei Noah's Ark Lab
[3]The Chinese University of Hong Kong [4]Huawei International Pte Ltd

Abstract

We present a simple and effective framework, named Point2Seq, for 3D object detection from point clouds. In contrast to previous methods that normally predict attributes of 3D objects all at once, we expressively model the interdependencies between attributes of 3D objects, which in turn enables a better detection accuracy. Specifically, we view each 3D object as a sequence of words and reformulate the 3D object detection task as decoding words from 3D scenes in an auto-regressive manner. We further propose a lightweight scene-to-sequence decoder that can auto-regressively generate words conditioned on features from a 3D scene as well as cues from the preceding words. The predicted words eventually constitute a set of sequences that completely describe the 3D objects in the scene, and all the predicted sequences are then automatically assigned to the respective ground truths through similarity-based sequence matching. Our approach is conceptually intuitive and can be readily plugged upon most existing 3D-detection backbones without adding too much computational overhead; the sequential decoding paradigm we proposed, on the other hand, can better exploit information from complex 3D scenes with the aid of preceding predicted words. Without bells and whistles, our method significantly outperforms previous anchor- and center-based 3D object detection frameworks, yielding the new state of the art on the challenging ONCE dataset as well as the Waymo Open Dataset. Code is available at https://github.com/ocNflag/point2seq.

1. Introduction

3D object detection is a critical component of intelligent perception systems for self-driving, aiming to localize and recognize cars, pedestrians, and other key objects around an autonomous vehicle. With the increasing popularity of LiDAR sensors, 3D object detection from point clouds has been receiving much attention owing to the advanced detection accuracy compared to other input modalities.

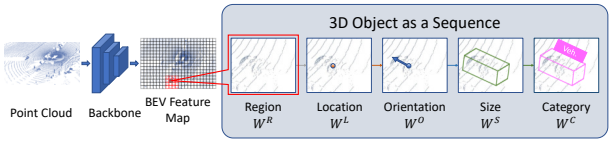

Figure 1. Point2Seq reformulates the 3D object detection problem as auto-regressively generating sequences of words that can represent the 3D objects. The sequential decoding paradigm attains better detection performance compared to previous works that predict all attributes of the 3D objects in parallel, thanks to its competence in exploring intrinsic dependencies among object attributes.

In driving scenes, most 3D objects are extremely small relative to the detection range, and numerous approaches have been employed to detect those small objects from point clouds accurately. Anchor-based methods [13, 38] place predefined anchors on each pixel center of a Bird-Eye-View (BEV) feature map, while center-based approaches [9, 42] treat the pixels near object centers as positives and predict boxes using those pixel features. These methods rely on complex hand-crafted procedures of label assignments and post-processing, and quantization errors introduced by the BEV representations lead to severe misalignment between the object locations and the pixel features used to predict those objects. Approaches like [16, 25, 27] rely on a second refinement stage to mitigate the misalignment issue, yet at the cost of adding too much computational overhead. Therefore, learning more spatially-aligned features to detect the 3D objects accurately while maintaining a high efficiency poses an open challenge to the research community.

To address the challenge, in this paper, we introduce Point2Seq, a flexible and streamlined framework for 3D object detection from point clouds. Unlike prior methods that typically predict all attributes of a 3D object (*e.g.*, location, class, size) simultaneously, we represent each object as a sequence, in which each word corresponds to an object attribute, and we explicitly explore the inherent dependencies among words alongside their relations with the input 3D scene to progressively predict each attribute of 3D objects. Our motivation is quite intuitive: given the fact that each ob-

*Joint first authors with equal contribution.
†Corresponding author. Email: xinchao@nus.edu.sg

ject is represented by sequential words, the existing words will provide 3D detectors with cues to better exploit spatial features and help detectors predict the following words more accurately. For instance, 3D detectors can leverage more spatially-aligned features for an object if the object location has been formerly predicted and better recognize the object class if its size information has already been known. It is therefore desirable to design a detection framework that may sequentially predict words of 3D objects conditioned on the preceding generated words as well as the spatial features until all the words form a set of sequences describing the 3D objects in the scene.

To achieve this goal, we must address two critical challenges: how to design the sequential object words prediction module and make it compatible with the existing 3D detection pipelines and how to optimize the 3D detector with the ground truth and predicted sequences. To resolve the first problem, we propose a novel scene-to-sequence decoder, which takes the BEV feature map and a set of initial region cues as input and auto-regressively decodes sequences for all objects in parallel. The scene-to-sequence decoder is compatible with most grid-based 3D backbones [13, 19, 35, 38, 44] and can effectively aggregate features from those backbones based on the information of the preceding words. By virtue of the highly-parallel deep learning libraries, the scene-to-sequence decoder can generate the sequences of all 3D objects in one shot, with little added time and memory cost.

To handle the second issue, we adopt the set-to-set loss to match the predicted sequences with the ground truths. Unlike existing approaches [3, 21, 36] that utilize the sum of the classification and regression loss as the cost function for bipartite matching, in this paper, we propose a novel metric to measure the similarity between two sequences. Then we perform bipartite matching by maximizing the global similarity of the prediction and the ground truth set using the proposed metric. In this manner, each predicted sequence can be automatically assigned to a respective ground truth without pre-defined anchors or centers. The assignments are globally optimal and result in better performance compared to previous methods.

With the lightweight scene-to-sequence decoder, our method can progressively predict the words of 3D objects, yielding reliable predictions that significantly outperform state-of-the-art. In addition, our method is free from the human-designed procedures of label assignments with similarity-based sequence matching. Our key contributions are summarized as follows:

- We present an effective and flexible framework for 3D object detection from point clouds. We represent each 3D object as a sequence of words and model the 3D object detection problem as decoding the words from the 3D scenes in an auto-regressive manner.
- We propose a scene-to-sequence decoder that can auto-regressively generate sequences representing the detected 3D objects and introduce the similarity-based sequence matching scheme to enable automatic assignments of the predicted sequences to the respective ground truths for end-to-end training.
- Our method significantly outperforms the anchor-based and center-based 3D detectors with the same backbone, attaining 66.16% mAP on the ONCE dataset and 77.52% vehicle L1 mAP on the Waymo Open Dataset.

2. Related Work

Backbones for 3D object detection. 3D detectors rely on various backbone networks to extract features from input point clouds. The existing backbones of 3D detectors can be divided into 3 streams: point-based, range-based, and grid-based. The point-based backbones [15, 22, 26, 28, 39–41] operate directly on raw point clouds with the point operators [18, 23] to extract the point-wise features. The range-based backbones [1, 8, 14, 20, 31] take range images, which are the raw data from LiDAR sensors, as the input representation. Customized operators, *e.g.*, range conditional convolutions [1], meta-kernels [8], are applied on the range images for feature extraction. The grid-based backbones [13, 19, 35, 38, 44] firstly rasterize point clouds into voxels or pillars. Those voxels or pillars are fed into a 3D network and then projected into a BEV feature map, followed by a 2D convolutional neural network to detect 3D objects. Among the 3 kinds of backbones, the grid-based backbones can obtain superior detection performance while maintaining high efficiency. Our Point2Seq is a flexible detection framework and can be applied to most grid-based backbones.

3D objects prediction mechanisms. 3D detectors adopt various prediction mechanisms to generate detected 3D objects from the backbone features. For the point-based backbones, PointRCNN [26] directly generates object proposals on the key points' locations. For the range-based backbones, RangeDet [8] generates 3D bounding boxes on the pixels of range images. For the grid-based backbones, SECOND [38] places a set of 3D anchors on every grid center of a BEV map. The anchors that have a high overlap with the ground truth 3D objects are set to positives, and the objects are then predicted on the positive anchors. SA-SSD [11] applies the part-sensitive warping scheme to enhance the spatial features. CenterPoints [42] treats BEV pixels near the object centers as positives and generates predicted bounding boxes near the object centers. Existing methods usually predict all attributes of the 3D objects simultaneously without considering the intra-object information, while Point2Seq can model the interdependencies among the object's attributes with the scene-to-sequence decoder.

Set-to-set matching for object detection. The set-to-set matching mechanism is first introduced in DETR [3] in

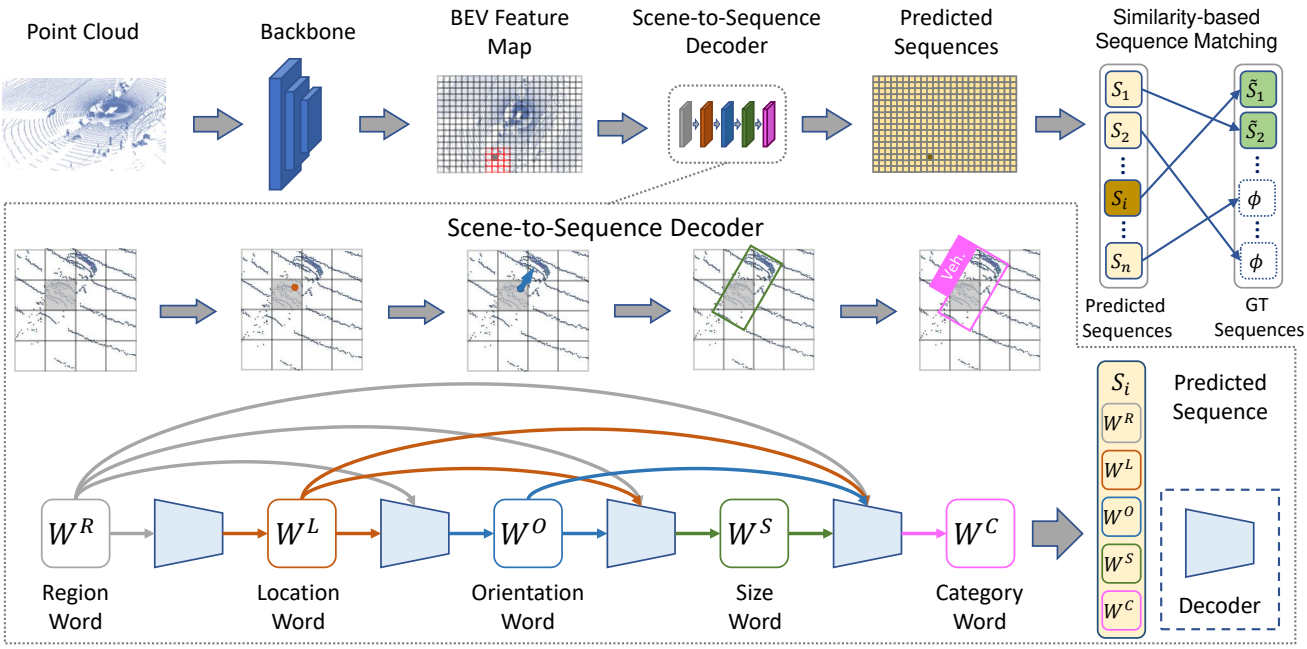

Figure 2. The overall architecture of our framework. Point2Seq contains three major components: the 3D backbone, the scene-to-sequence decoder, and the similarity-based sequence matching scheme. The 3D backbone takes the rasterized point cloud as input and outputs the Bird-Eye-View (BEV) feature map for the 3D scene. The scene-to-sequence decoder operates on the BEV feature map and sequentially predicts the words of 3D objects based on the information from the preceding predicted words. Finally, the predicted sequences are automatically assigned to the corresponding ground truths by the proposed similarity-based sequence matching scheme.

which a set of object queries are assigned to the respective ground truths through bipartite matching. This method is improved by a series of works [29, 32, 34, 45] on image-based object detection. The set-to-set matching scheme has also been adopted in 3D object detection. 3DETR [21] utilizes sampled points as object queries. Object DGCNN [36] leverages a sparse set of object queries to iteratively interact with a BEV feature map to generate 3D objects. Compared to previous methods, Point2Seq does not require the sparse object queries or the multi-step refinement modules in [33, 37]. The sequences are generated densely from the BEV map in parallel and are automatically matched to the corresponding ground truths by similarity-based sequence matching, without additional modules or parameters.

3. Detecting 3D Objects as Sequences

3.1. Architecture

For each 3D scene, Point2Seq takes a point cloud as input and outputs a set of 3D bounding boxes $\mathcal{B} = \{B_1, \cdots, B_M\} \in \mathbb{R}^{M \times 8}$ that represent the detected 3D objects, *e.g.*, vehicles, pedestrians, cyclists, *etc*. A 3D point cloud is an $N \times d$ matrix, where N denotes the number of points in the scene and d denotes the initial features of points, *i.e.*, 3D coordinates, intensity, *etc*. Each 3D object $B_i \in \mathbb{R}^8$ is a vector: $[x, y, z, l, w, h, \theta, c]$, where $[x, y, z]$ is the location of the object's center, $[l, w, h]$ is the object's size, θ is the object's orientation, and c is the class of the object.

As is shown in Figure 2, the architecture of Point2Seq is composed of 3 parts: the 3D backbone, the scene-to-sequence decoder, and the similarity-based sequence matching scheme. The 3D backbone first consumes a point cloud and generates a BEV feature map from the point cloud. Then the scene-to-sequence decoder takes both the BEV feature map and an initial set of region cues as input and decodes sequences of words that describe the detected 3D objects in the scene. Finally, the similarity-based sequence matching is applied to assign the predicted sentences to the respective ground truths. The choice of 3D backbones in Point2Seq can be flexible: most grid-based backbones [13, 19, 35, 38] can be applied in our framework. The grid-based backbones first transform point clouds into voxels or pillars, and then 3D features are extracted from those voxels or pillars by sparse convolutions [10] or set abstraction [23], respectively. The 3D features are then projected to Bird-Eye-View (BEV). A 2D convolutional neural network is applied on the projected features to obtain the final BEV feature map $F \in \mathbb{R}^{H \times W \times C}$, where the detection space is divided into an $H \times W$ grid, and C denotes the number of feature channels.

In the scene-to-sequence decoder, the 3D objects $\mathcal{B}$ are transformed into a set of sequences $\{S_1, \cdots, S_M\}$, where each sequence S_i corresponds to a 3D object B_i and con-

tains K words $\{W_i^0, \cdots, W_i^{K-1}\}$ that can represent the 3D object. The scene-to-sequence decoder operates on the BEV feature map F and can auto-regressively predict a word W_i^j conditioning on F and the preceding words $W_i^{0:j-1}$. We will introduce the detailed design of the scene-to-sequence decoder in Sec. 3.2 and then illustrate how to optimize the scene-to-sequence decoder and the 3D backbone through similarity-based sequence matching in Sec. 3.3. Finally, we discuss and compare our method with previous literature in Sec. 3.4.

3.2. Scene-to-Sequence Decoder

Problem formulation. Previous single-stage 3D detectors model the 3D object detection problem as predicting all attributes of the 3D objects $\mathcal{B}$ simultaneously from the features F. The learning process can be formulated as the optimization problem:

$$\max \sum_{i=1}^{M} logP(B_i|\mathcal{D}(F)), \tag{1}$$

where $B_i \in \mathcal{B}$ is the attributes $[x, y, z, l, w, h, \theta, c]$ of the ith ground truth 3D object, $M = |\mathcal{B}|$, and $\mathcal{D}$ is normally a convolutional prediction head applied on the BEV feature map F. The parallel prediction paradigm is widely adopted for its efficiency. However, it suffers from the misalignment between the object's actual location and the BEV features used for prediction due to the high quantization errors introduced by rasterization. Multi-stage refinements can be employed to mitigate the misalignment problem but will consequently introduce too much computational overhead.

In this paper, we draw inspirations from Language Modeling (LM) [2, 7] in natural language processing applications and translate each 3D object B into a sequence S containing K words $\{W^0, \cdots, W^{K-1}\}$:

$$B = \mathcal{T}(S) = \mathcal{T}(W^0, \cdots, W^{K-1}). \tag{2}$$

The translation $\mathcal{T}$ is parameter-free and bidirectional, so the 3D objects and their corresponding words can be easily transformed into each other. Then similar to the language models, we can reformulate the detection problem as maximizing the probability production of all target words $\tilde{W_i}^j$ conditioning on the feature F and the preceding predicted words $W_i^{0:j-1}$:

$$\max \sum_{i=1}^{M} \sum_{j=1}^{K-1} logP(\tilde{W_i}^j|\mathcal{D}(F, W_i^0, \cdots, W_i^{j-1})). \tag{3}$$

The main insight of our approach is that each 3D object is decomposed into several words, and predicting these words sequentially, instead of simultaneously in previous methods, enables more effective exploitation of the BEV features with the cues from the preceding predicted words.

3D object as a sequence of words. Translating every 3D object into words is a pivotal step in our method. Different from those methods that adopt discrete tokens as words in natural language processing tasks, we represent the words in a continuous format in our method. The use of continuous representations for object words is preferable in 3D object detection since most attributes of 3D objects, *e.g.*, locations, sizes, orientations, are continuous values, and the predicted words can be directly transformed back into the corresponding object's attributes without loss of accuracy.

In this paper, each 3D object $B = [x, y, z, l, w, h, \theta, c]$ is translated into 5 words:

$$B = \mathcal{T}(S) = \mathcal{T}(W^R, W^L, W^O, W^S, W^C). \tag{4}$$

The region word $W^R = [R_x, R_y] \in \mathbb{R}^2$ indicates the possible region in which the 3D object is likely to appear, where $[R_x, R_y]$ is the BEV center coordinate of the region, and additional parameters $[R_l, R_w]$ are introduced to describe the spatial range of the region on the BEV feature map. The location word $W^L = [L_x, L_y, z] \in \mathbb{R}^3$ denotes the location of the object's center, where $L_x = (x - R_x)/R_l$ and $L_y = (y - R_y)/R_w$ denote the relative location in the region. The orientation word $W^O = [sin(\theta), cos(\theta)] \in \mathbb{R}^2$ encodes the object's orientation θ by trigonometric functions. The size word $W^S = [log(l), log(w), log(h)] \in \mathbb{R}^3$ applies logarithmic functions to the object's size. The category word $W^C \in \mathbb{R}^{n+1}$ indicates the probabilities of n detected classes and the background class.

Scene-to-sequence prediction. Our proposed scene-to-sequence decoder takes the BEV features F and a set of region words W^R as the initial inputs and sequentially predicts W^L, W^O, W^S, W^C, *i.e.*, W^1, W^2, W^3, W^4 in 4 steps for each region W^R. In each step, the words are predicted on a hidden state map $H \in \mathbb{R}^{H \times W \times C}$ that encodes the historical information of the preceding steps. Specifically, the hidden state H_1 is firstly initialized as the input BEV feature map F:

$$H_1 = F. \tag{5}$$

Then at the jth step, the word W^j will be directly predicted from the hidden state H_j at the corresponding region center W^R, *i.e.*, $H_j[W^R] \in \mathbb{R}^C$, through a single linear projection layer f_{linear}:

$$W^j = f_{linear}(H_j[W^R]), \tag{6}$$

where $[\cdot]$ is the indexing operator. The hidden state $H_j[W^R]$ at W^R will then be updated to the next step $H_{j+1}[W^R]$ based on the already learned knowledge from the former predicted words $\{W^0, \cdots, W^j\}$:

$$H_{j+1}[W^R] = \Phi(H_j[W^R]; W^0, \cdots, W^j), \tag{7}$$

where Φ is the update function. To model the hidden state update process at the jth step, near each region W^R, we

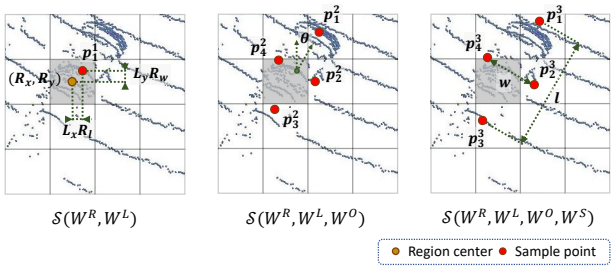

Figure 3. The sample locations at each step. For each region, we progressively obtain 1, 4, 4 points from the spatial sampler $\mathcal{S}$, based on the information from the predicted words.

first sample a sparse set of points $\{p_1^j, \cdots, p_n^j\} \in \mathbb{R}^{n\times 2}$ by a spatial sampler $\mathcal{S}$ parameterized by the predicted words:

$$\{p_1^j, \cdots, p_n^j\} = \mathcal{S}(W^0, \cdots, W^j). \tag{8}$$

The sampling patterns of $\mathcal{S}$ are shown in Figure 3, and the detailed formulations are demonstrated in the appendix. Then for each region W^R, we can update to $H_{j+1}[W^R]$ by aggregating hidden vectors at those sampled locations on H_j, which can be formulated as

$$H_{j+1}[W^R] = \mathcal{A}(H_j[p_1^j], \cdots, H_j[p_n^j]), \tag{9}$$

where the aggregation function $\mathcal{A}$ concatenates the sampled hidden vectors and projects them into the $\mathbb{R}^C$ space.

The initial set of the region words W^R indicates where the 3D objects are likely to appear in the 3D scene. Since there is no such prior information, we employ a dense prediction strategy in this paper. Namely, we treat each pixel in the BEV feature map as a region word and predict a sequence for each BEV pixel. The category word W^C is expected to predict the highest probability for the background class if there is no 3D object near the corresponding pixel region. The dense prediction paradigm benefits from the highly-parallel characteristics of the modern deep learning libraries, and the sequences for all pixels can be predicted in parallel by the scene-to-sequence decoder efficiently using shared MLPs and sampling operators.

3.3. Similarity-based Sequence Matching

In this section, we will introduce how to optimize our detection framework by similarity-based sequence matching. Our approach is inspired by the set-to-set loss employed in the image-based detection frameworks [3]. Our main contribution lies in the design of a new cost function tailored for 3D object detection in the set-to-set matching problem. Specifically, we propose a novel metric $Sim(S, \tilde{S})$ to measure the similarity of two sequences S and $\tilde{S}$ from the predicted and the ground truth sequence set, respectively. We match the predicted sequences to the corresponding target sequences by maximizing the global similarity of the two sequence sets. Finally, losses can be applied to the matched sequence pairs for back-propagation. In this manner, we can eliminate the hand-crafted label assignment process and make our model end-to-end trainable without non-maximum suppression.

The scene-to-sequence decoder outputs a set of predicted sequences $\{S_1, \cdots, S_{HW}\}$ containing $H \times W$ sequences for all BEV pixels in total. For the ground truth 3D objects, we also construct a sequence set $\{\tilde{S}_1, \cdots, \tilde{S}_M, \varnothing, \cdots, \varnothing\}$ where the size of the ground truth set is equal to that of the prediction set, and we pad the remaining sequences with $\varnothing$ if $M < H \times W$. To measure the similarity between a predicted sequence S and a ground truth sequence $\tilde{S}$, we define a new similarity metric that can be formulated as

$$Sim(S, \tilde{S}) = (W^C \tilde{W}^C)^{\alpha} \cdot e^{-(1-\alpha)\sum_{j\in\{R,L,O,S\}} |W^j - \tilde{W}^j|}, \tag{10}$$

where the first term $(W^C \tilde{W}^C)^{\alpha}$ measures the class similarity between the predicted and the ground truth objects, and the second term measures the shape and location similarity. The hyper-parameter α is utilized to balance the two similarities and set to 0.25 in our experiments. $Sim(S, \varnothing) = 0$ if a predicted sequence is matched to $\varnothing$.

The proposed similarity metric is a more stringent criterion to match the predictions with the ground truths, and even slight differences between the two sequences can make the similarity score tend to 0. Given the fact that the 3D objects are innately small and the mismatch should be completely avoided, the stringent similarity metric we proposed is preferable in the task of 3D object detection.

With the similarity metric, we can further establish the optimal set-to-set matching Π^* by considering the bipartite matching problem:

$$\Pi^* = \arg\max_{\Pi} \sum_{(i\to j)\in\Pi} Sim(S_i, \tilde{S}_j), \tag{11}$$

where Π is a bijective function that enables a one-to-one mapping from the predicted sequence set to the ground truth set. The bipartite matching problem aims to find the optimal Π^* so that the maximum overall similarity of the two sets can be achieved. With Π^*, every ground truth sequence can be automatically assigned to the corresponding predicted sequence that has the highest similarity. The optimal bipartite matching Π^* can be calculated efficiently by the Hungarian algorithm [12].

Once the matched pairs of S and $\tilde{S}$ are established, the proposed loss function tailored for 3D object detection can be computed as:

$$\mathcal{L}_{det} = \sum_{(i\to j)\in\Pi^*} [\mathcal{L}_{cls}(W_i^C, \tilde{W}_j^C) + \mathbb{1}_{\{\tilde{S}_i \neq \varnothing\}} \lambda_{reg} \mathcal{L}_{reg}(W_i^{\{R,L,O,S\}}, \tilde{W}_j^{\{R,L,O,S\}})], \tag{12}$$

where $\mathcal{L}_{cls}$ is the focal loss applied on the predicted and target category words, and $\mathcal{L}_{reg}$ takes the words $W^{\{R,L,O,S\}}$

and $\tilde{W}^{\{R,L,O,S\}}$ as input and translates them back to the respective object's location and shape $[x, y, z, l, w, h, \theta]$ on which the smooth-L_1 loss is then applied. The indicator function $\mathbb{1}_{\{\tilde{S}_i \neq \varnothing\}}$ implies that we only apply $\mathcal{L}_{reg}$ on those sequences that are matched to the ground truths, and λ_{reg} is a coefficient that balances the two losses.

Since each ground truth 3D object is matched with only one predicted sequence, the scene-to-sequence head does not produce duplicated boxes for an individual object. Hence the time-consuming process of non-maximum suppression can be eliminated in our framework. During the inference stage, we simply filter out those low-quality sequences in which the maximum class probability in W^C is below a certain threshold, and we translate the remaining sequences into 3D objects as the final detection results.

3.4. Discussion

Our proposed Point2Seq shares a similar intuition with the concurrent work Pix2Seq [5], which is proposed for image-based object detection, in terms of leveraging objects as words that can be read out from a feature map. However, our method is intrinsically different from [5] in 3 aspects: 1) Unlike [5] that merges all objects into an individual sequence, we treat each object as a sequence and predict all objects in parallel, while words in each object are generated sequentially. In this manner, we can circumvent the object ordering problem in [5], and our method is much more efficient at the inference stage compared to [5] in which the inference latency will be heavily influenced by the total object count in an image. 2) We adopt the continuous word representations instead of discrete tokens in [5]. The use of continuous representations relieves the need for quantization and makes our method compatible with the existing loss functions tailored for 3D object detection. 3) We propose the scene-to-sequence decoder to generate words for each object, in lieu of the Transformer architecture in [5]. The scene-to-sequence decoder is lightweight and leverages a sparse set of features to predict each object, which is more suitable for 3D object detection where the detected targets usually are small and sparse.

4. Experiment

In this section, we evaluate Point2Seq on the commonly-used Waymo Open Dataset [30] and the ONCE dataset [17]. We first introduce the experimental settings in Sec. 4.1. Then we compare our approach with previous state-of-the-art methods on the Waymo Open Dataset (Sec. 4.2) and the ONCE dataset (Sec. 4.3). Finally, we report the inference speed and the number of parameters, as well as the efficacy of different components in our model in Sec. 4.4.

4.1. Experimental Setup

Waymo Open Dataset. The Waymo Open Dataset is composed of 1000 sequences of point clouds, in which 798 sequences (nearly $158k$ point cloud samples) are used as the training set, and 202 sequences (nearly $40k$ point cloud samples) are utilized as the validation set. The evaluation metrics on the Waymo Open Dataset are 3D mean Average Precision (mAP) and mAP weighted by heading accuracy (mAPH). The IoU threshold used for vehicles is 0.7 and 0.5 for other categories. The detection results are reported based on the difficulty levels: LEVEL 1 for boxes with more than 5 points and LEVEL 2 for boxes with at least 1 point.

ONCE Dataset. The ONCE dataset contains one million point clouds in total, in which 5k, 3k, 8k point clouds are annotated as the training, validation, testing split, respectively. The remaining point clouds are kept unannotated for self-/semi-supervised learning. In this paper, we train our model on the training split and report the detection results of vehicles, pedestrians, and cyclists on the validation and testing split, without using the unlabeled data. The official evaluation metric is mean Average Precision (mAP), and the detection results are divided according to the objects' distances to the sensor: 0-30m, 30-50m, and 50m-inf.

Implementation Details. On the Waymo Open Dataset, we use the same 3D sparse convolutional neural network and 2D convolutional neural network as [42]. The input voxel size and the output resolution of the BEV feature map are also kept the same as [42] for a fair comparison. On the ONCE dataset, all the voxel-based detectors use the same type of 3D backbone [38] in their official benchmark implementations. We also follow the setting and use the 3D backbone in [38]. For other model configurations, we adopt the same as those on the ONCE benchmark.

Training and Inference Details. We train our model with the ADAM optimizer and the cosine annealing learning rate scheduler. On the Waymo Open Dataset, we uniformly sample 20% of the point cloud samples for training and use the full validation set for evaluation following [25]. We train our model with the batch size 32 and the initial learning rate 0.006 for 180 epochs on 8 V100 GPUs. λ_{reg} in the loss function is set to 2. Data augmentations are kept the same as [42]. On the ONCE dataset, we follow the training settings of the respective benchmark and train our model with the batch size 32 and the initial learning rate 0.003 for 80 epochs on 8 V100 GPUs. λ_{reg} in the loss function is set to 0.5. Data augmentations are kept the same as [17]. On both two datasets, we filter out those objects with the maximum foreground class probability in W^C below 0.2 and keep the remaining objects as the final detection results during the inference stage, without any other post-processing.

4.2. Comparisons on the Waymo Open Dataset

Since our contribution focuses on the 3D object prediction mechanism, the fairest way to evaluate our method and compare it with the anchor-based and center-based methods is to only replace the center or anchor head with our scene-to-sequence decoder while maintaining other components

Method	Backbone	Head	Vehicle LEVEL 1		Vehicle LEVEL 2	
			3D mAP(%)	3D mAPH(%)	3D mAP(%)	3D mAPH(%)
LaserNet [20]	Range	Anchor	52.1	50.1	-	-
RCD [1]	Range	Center	69.0	68.5	-	-
RangeDet [8]	Range	Center	72.85	-	-	-
RSN [31]	Range	Center	75.1	74.6	66.0	65.6
PointPillars [13]	Pillar	Anchor	63.3	62.7	55.2	54.7
Pillar-OD [35]	Pillar	Anchor	69.8	-	-	-
MVF [43]	Voxel	Anchor	62.93	-	-	-
PV-RCNN [25]	Voxel	Anchor	77.51	76.89	68.98	68.41
VoTr-TSD [19]	Voxel	Anchor	74.95	74.25	65.91	65.29
Voxel R-CNN [6]	Voxel	Anchor	75.59	-	66.59	-
Pyramid-RCNN [16]	Voxel	Anchor	76.3	75.68	67.23	66.68
CT3D [24]	Voxel	Anchor	76.3	-	69.04	-
CVCNet [4]	Voxel	Center	65.20	-	-	-
AFDet [9]	Voxel	Center	63.69	-	-	-
CenterPoints [42]	Voxel	Center	76.7	76.2	68.8	68.3
SECOND† [38]	Voxel	Anchor	73.62	73.14	64.86	64.40
CenterPoints† [42]	Voxel	Center	75.58	75.01	67.00	66.52
Point2Seq (Ours)	**Voxel**	**Sequence**	**77.52**	**77.03**	**68.80**	**68.36**

Table 1. Performance comparison on the Waymo Open Dataset with 202 validation sequences for vehicle detection. †: re-implemented using the official code. Point2Seq maintains the same backbone, data augmentations, and training epochs with the re-implemented baselines.

Method	mAP(%)	Vehicle mAP(%)				Pedestrian mAP(%)				Cyclist mAP(%)			
		overall	0-30m	30-50m	50m-inf	overall	0-30m	30-50m	50m-inf	overall	0-30m	30-50m	50m-inf
PointRCNN [25]	28.74	52.09	74.45	40.89	16.81	4.28	6.17	2.4	0.91	29.84	46.03	20.94	5.46
PointPillars [13]	44.34	68.57	80.86	62.07	47.04	17.63	19.74	15.15	10.23	46.81	58.33	40.32	25.86
PV-RCNN [25]	53.55	77.77	89.39	72.55	58.64	23.50	25.61	22.84	17.27	59.37	71.66	52.58	36.17
SECOND [38]	51.89	71.19	84.04	63.02	47.25	26.44	29.33	24.05	18.05	58.04	69.96	52.43	34.61
CenterPoints [42]	60.05	66.79	80.10	59.55	43.39	49.90	56.24	42.61	26.27	63.45	74.28	57.94	41.48
Point2Seq (Ours)	**66.16**	**73.43**	**85.16**	**66.21**	**50.76**	**57.53**	**68.21**	**47.15**	**25.18**	**67.53**	**77.95**	**62.14**	**46.06**

Table 2. Performance comparison on the ONCE dataset validation split. Point2Seq maintains the same backbone architecture and training configurations with the baselines on the ONCE benchmark.

as the same. We follow this principle and re-implement two baseline models from their official implementations: SECOND [38] with the anchor head and CenterPoints [42] with the center head. Our proposed Point2Seq, re-implemented SECOND, and CenterPoints have the same voxel-based 3D backbone, data augmentations, and training epochs to ensure a completely fair comparison.

Table 1 shows the detection results on the Waymo validation set. Simply switching from the anchor and center head to our Point2Seq gives 3.90% and 1.94% LEVEL 1 mAP improvements, respectively. Our method attains 77.52% LEVEL 1 mAP and 68.80% LEVEL 2 mAP for vehicle detection, surpassing existing methods by a significant margin. Our approach even outperforms those time-consuming two-stage 3D detectors [16, 24, 25], which indicates the effectiveness of the scene-to-sequence decoder.

4.3. Comparisons on the ONCE dataset

The ONCE dataset benchmarks different voxel-based detectors using the same backbone network, and we also follow this rule for a fair comparison. As is shown in Table 2, Point2Seq attains the state-of-the-art results on all classes, with 73.43% mAP for vehicle detection, 57.53% mAP for pedestrian detection, and 67.53% for cyclist detection. The overall mAP of our approach is 66.16%, 6.11% higher than the center-based 3D object detector [42] and 14.27% higher than the anchor-based 3D object detector [38]. The observations on the ONCE dataset are consistent with those on the Waymo Open Dataset.

4.4. Ablation Studies

Inference speed and model parameters. Table 3 demonstrates the inference time and the number of parameters of our method. Since the 3D objects in a scene are predicted in parallel, Point2Seq can obtain high efficiency with 70.4ms inference latency on average for a single model on a V100 GPU. The scene-to-sequence head only contains several linear projection layers, and the sampling operation is parameter-free, so the model only introduces 0.1M additional parameters compared to the center-based baseline.

Method	Vehicle L1/L2 mAP(%)	#Param	Latency (ms)
PV-RCNN [25]	77.51/68.98	13.05M	300
CenterPoints [42]	76.7/68.8	8.74M	77
SECOND† [38]	73.62/64.86	7.28M	66.5
CenterPoints† [42]	75.58/67.00	7.76M	69.5
Point2Seq† (Ours)	77.52/68.80	7.86M	70.4

Table 3. Inference speed and parameters amount. †: tested under the same environment using a single model on a V100 GPU.

Backbone	Head	Veh. LEVEL 1 mAP/mAPH(%)	Veh. LEVEL 2 mAP/mAPH(%)
Pillar	Anchor	63.31/62.74	55.24/54.72
	Center	65.46/64.66	57.59/56.88
	Point2Seq	**69.01/68.25**	**60.72/60.03**
Voxel	Anchor	73.62/73.14	64.86/64.40
	Center	75.58/75.01	67.00/66.52
	Point2Seq	**77.52/77.03**	**68.80/68.36**

Table 4. Performances on different backbone networks.

Generalizability on different 3D backbones. To verify whether Point2Seq can achieve superior performances upon different backbones, we apply the scene-to-sequence head on both the voxel-based [42] and pillar-based [13] 3D backbones and compare the results with the center and anchor head, respectively. Table 4 demonstrates that on both two types of backbone networks, our method consistently outperforms the anchor-based and center-based detectors.

Effects of different components in Point2Seq. Table 5 shows the effectiveness of the scene-to-sequence decoder and the similarity-based sequence matching scheme. Similarity-based sequence matching can be independently applied on the previously used convolutional head and boost the detection performance by 3.1% mAP compared to the anchor-based baseline. Combing the two proposed components, we can obtain a performance gain of 3.9% mAP.

The order of words in a sequence. We explored the influence of changing the predicted word orders in our method. The results in Table 6 indicate that the order of words plays a non-negligible role on the detection performance. For example, putting W^O at the 4th place greatly reduces the detection accuracy, which may indicate the importance of predicting the object orientation at an earlier position. Putting W^C at the end will exhibit better performance compared to putting W^C at the beginning.

The choice of different similarity metrics. We evaluated different formulas of the similarity metric used in similarity-based sequence matching. Table 7 exhibits the results of the 3 formulas we have examined. The formula (3) is the currently adopted similarity metric. The formula (2) replaces the term $e^{-(1-\alpha)\sum_{j\in\{R,O,S,L\}}|W^j-\tilde{W}^j|}$ in Eq. 10 with $e^{-(1-\alpha)3DIoU(B,\tilde{B})}$ where $3DIoU(B,\tilde{B})$ computes the 3D IoU score of two bounding boxes. The formula (1) replaces the same term with $e^{-(1-\alpha)\sum_{j=1}^{8}|C^j-\tilde{C}^j|}$, where we calculate the differences of 8 respective corners C of two bounding boxes. The results indicate that Eq. 10 is the best among the 3 formulas as the similarity metric.

Assignment	Module	Veh. LEVEL 1 mAP/mAPH(%)	Veh. LEVEL 2 mAP/mAPH(%)
Anchor	C.H	73.62/73.14	64.86/64.40
Center	C.H	75.58/75.01	67.00/66.52
S.S.M.	C.H	76.72/76.19	68.02/67.54
S.S.M.	S.S.D	**77.52/77.03**	**68.80/68.36**

Table 5. Effects of different components in Point2Seq. C.H: Convolutional Head in previous works. S.S.M.: Similarity-based Sequence Matching. S.S.D: Scene-to-Sequence Decoder.

Order	Veh. LEVEL 1 mAP/mAPH(%)	Veh. LEVEL 2 mAP/mAPH(%)
W^R, W^O, W^S, W^L, W^C	77.40/76.91	68.71/68.27
W^R, W^O, W^L, W^S, W^C	77.43/76.94	68.74/68.30
W^R, W^L, W^O, W^S, W^C	77.52/77.03	68.80/68.36
W^R, W^L, W^S, W^O, W^C	73.82/73.24	66.16/65.61
W^R, W^C, W^L, W^O, W^S	75.96/75.41	67.26/66.76
W^R, W^C, W^S, W^O, W^L	74.52/73.99	66.86/66.35
W^R, W^C, W^O, W^L, W^S	77.05/76.54	68.37/67.90
W^R, W^C, W^O, W^S, W^L	76.82/76.28	68.11/67.62

Table 6. Effects of different word orders in the sequences.

Metrics	Veh. LEVEL 1 mAP/mAPH(%)	Veh. LEVEL 2 mAP/mAPH(%)
(1)	74.31/73.52	65.61/65.04
(2)	75.07/74.43	66.36/65.73
(3)	**75.40/74.81**	**66.76/66.23**

Table 7. Comparisons of different similarity metrics.

5. Conclusion

We present Point2Seq, an effective and general 3D object detection framework that can be applied to most grid-based backbone networks. Point2Seq contains a scene-to-sequence decoder, which can auto-regressively generate sequences describing the detected 3D objects, and similarity-based sequence matching is proposed to enable end-to-end training without human-designed label assignments. For future works, we plan to extend our framework to multi-modality 3D object detection.

Acknowledgement

This work is supported by NUS Faculty Research Committee Grant (WBS: A-0009440-00-00), and NRF Centre for Advanced Robotics Technology Innovation (CARTIN) Project.

References

[1] Alex Bewley, Pei Sun, Thomas Mensink, Dragomir Anguelov, and Cristian Sminchisescu. Range conditioned dilated convolutions for scale invariant 3d object detection. *arXiv preprint arXiv:2005.09927*, 2020. 2, 7

[2] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 4

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. 2, 5

[4] Qi Chen, Lin Sun, Ernest Cheung, and Alan L Yuille. Every view counts: Cross-view consistency in 3d object detection with hybrid-cylindrical-spherical voxelization. *Advances in Neural Information Processing Systems*, 2020. 7

[5] Ting Chen, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection. *arXiv preprint arXiv:2109.10852*, 2021. 6

[6] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. *arXiv:2012.15712*, 2020. 7

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 4

[8] Lue Fan, Xuan Xiong, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Rangedet: In defense of range view for lidar-based 3d object detection. *arXiv preprint arXiv:2103.10039*, 2021. 2, 7

[9] Runzhou Ge, Zhuangzhuang Ding, Yihan Hu, Yu Wang, Sijia Chen, Li Huang, and Yuan Li. Afdet: Anchor free one stage 3d object detection. *arXiv preprint arXiv:2006.12671*, 2020. 1, 7

[10] Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9224–9232, 2018. 3

[11] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11873–11882, 2020. 2

[12] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 5

[13] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019. 1, 2, 3, 7, 8

[14] Zhidong Liang, Ming Zhang, Zehan Zhang, Xian Zhao, and Shiliang Pu. Rangercnn: Towards fast and accurate 3d object detection with range image representation. *arXiv preprint arXiv:2009.00206*, 2020. 2

[15] Huihui Liu, Yiding Yang, and Xinchao Wang. Overcoming catastrophic forgetting in graph neural networks. In *AAAI Conference on Artificial Intelligence*, 2021. 2

[16] Jiageng Mao, Minzhe Niu, Haoyue Bai, Xiaodan Liang, Hang Xu, and Chunjing Xu. Pyramid r-cnn: Towards better performance and adaptability for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2723–2732, 2021. 1, 7

[17] Jiageng Mao, Minzhe Niu, Chenhan Jiang, Hanxue Liang, Jingheng Chen, Xiaodan Liang, Yamin Li, Chaoqiang Ye, Wei Zhang, Zhenguo Li, et al. One million scenes for autonomous driving: Once dataset. *arXiv preprint arXiv:2106.11037*, 2021. 6

[18] Jiageng Mao, Xiaogang Wang, and Hongsheng Li. Interpolated convolutional networks for 3d point cloud understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1578–1587, 2019. 2

[19] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3164–3173, 2021. 2, 3, 7

[20] Gregory P. Meyer, Ankita Gajanan Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K. Wellington. Lasernet: An efficient probabilistic 3d object detector for autonomous driving. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12669–12678, 2019. 2, 7

[21] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2906–2917, 2021. 2, 3

[22] Xuran Pan, Zhuofan Xia, Shiji Song, Li Erran Li, and Gao Huang. 3d object detection with pointformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7463–7472, 2021. 2

[23] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2, 3

[24] Hualian Sheng, Sijia Cai, Yuan Liu, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, and Min-Jian Zhao. Improving 3d object detection with channel-wise transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2743–2752, October 2021. 7

[25] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020. 1, 6, 7, 8

[26] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 770–779, 2019. 2

[27] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 1

[28] Weijing Shi and Raj Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1711–1719, 2020. 2

[29] Peize Sun, Yi Jiang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. Onenet: Towards end-to-end one-stage object detection. *arXiv preprint arXiv:2012.05780*, 2020. 3

[30] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020. 6

[31] Pei Sun, Weiyue Wang, Yuning Chai, Gamaleldin Elsayed, Alex Bewley, Xiao Zhang, Cristian Sminchisescu, and Dragomir Anguelov. Rsn: Range sparse net for efficient, accurate lidar 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5725–5734, 2021. 2, 7

[32] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, et al. Sparse r-cnn: End-to-end object detection with learnable proposals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14454–14463, 2021. 3

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 3

[34] Jianfeng Wang, Lin Song, Zeming Li, Hongbin Sun, Jian Sun, and Nanning Zheng. End-to-end object detection with fully convolutional network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15849–15858, 2021. 3

[35] Yue Wang, Alireza Fathi, Abhijit Kundu, David A Ross, Caroline Pantofaru, Tom Funkhouser, and Justin Solomon. Pillar-based object detection for autonomous driving. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 18–34. Springer, 2020. 2, 3, 7

[36] Yue Wang and Justin Solomon. Object dgcnn: 3d object detection using dynamic graphs. *arXiv preprint arXiv:2110.06923*, 2021. 2, 3

[37] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 3

[38] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 1, 2, 3, 6, 7, 8

[39] Yiding Yang, Zunlei Feng, Mingli Song, and Xinchao Wang. Factorizable graph convolutional networks. In *Advances in Neural Information Processing Systems*, 2020. 2

[40] Yiding Yang, Jiayan Qiu, Mingli Song, Dacheng Tao, and Xinchao Wang. Distilling knowledge from graph convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 2

[41] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11040–11048, 2020. 2

[42] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11784–11793, 2021. 1, 2, 6, 7, 8

[43] Yin Zhou, Pei Sun, Yu Zhang, Dragomir Anguelov, Jiyang Gao, Tom Ouyang, James Guo, Jiquan Ngiam, and Vijay Vasudevan. End-to-end multi-view fusion for 3d object detection in lidar point clouds. In *Conference on Robot Learning*, pages 923–932. PMLR, 2020. 7

[44] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4490–4499, 2018. 2

[45] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 3