Point2Seq

Abstract

- point clouds input을 받아 3D object detection하는 Task를 auto-regressive 방식으로 words를 decode하는 Task로 해결함
  - 3D objects의 attributes간의 interdependency를 고려할 수 있음
  - light-weight scene-to-sequence decoder를 사용함
    - object가 parallel하게 뽑힘
      - parallel해서 similarity based sequence matching이 필요함
- 핵심은 Sequential decoder를 제안한 것임
  - 기존의 anchor-based, center-based 3D objet detection보다 성능이 좋음
- 코드는 pytorch로 공개
  - https://github.com/ ocNflag/point2seq

Introduction

- 3D object detection은 자율주행에 꼭 필요한 요소이고 LiDAR sensor를 많이 쓰니까 point cloud를 input으로 많이 받음 → 내가 하려는 Task가 중요함
- 주행에서 대부분의 3d object는 극도로 작고 많은 방법론이 제안되었음
  1. Anchor-based methods
     - predefined anchor를 BEV feature map의 pixel center마다 정해둠
  2. Center-based methods
     - object center주변의 pixel을 positive로 여기고 거기에 해당하는 pixel feature로 box를 예측
  - 이러한 방법론은 label assignment나 post-processing에서 복잡한 과정을 필요로 함

    - quantization error가 발생하기 쉬움, misalignment가 존재함
        - refine stage를 많이들 사용했음
- 유연하고 간소화된 Point2Seq를 제안함
    - location, size, class등을 progressively하게 predict해서 inherent dependency를 더 고려함
    - Existing words가 다음 word를 예측하는데 도움을 줄 수 있음
        - Region, Location, Orientation, Size, Category순으로 예측
        - 예를 들어, object location이 어느정도 주어지면 거기에 aligned된 Feature를 다음에 더 쓸 수 있고, size information이 category를 예측하는데 도움을 줄 수 있음
- Point2Seq를 구현하는데 크게 2가지 challenge가 있음
    1. 어떻게 sequential object words를 design해서 기존의 3D detection pipeline에 넣을 것인가?
        - BEV feature map과 initial region cue를 받고, auto-regressive하게 sequence를 뽑으면 됨
        - 단 objects는 parallel함
    2. 어떻게 3D detector가 predict한 sequence를 gt sequence로 optimize할 것인가?
        - set-to-set loss를 줌
            - 이전의 bipartite matching (loss를 최소화)과 다르게, similarity를 구하는 새로운 metric를 제안함
            - 어쨌든 bipartite matching으로 global similarity를 최대화 하는건 같음
            - 그래서 pre-defined anchor나 center없이 loss를 줄 수 있음 → 새로운 방식이 더 간단한게 맞나…?
- lightweight scene-to-sequence decoder로, progressively하게 words를 예측할 수 있음
    - label assignment할 때도 human design이 따로 필요없음

Related Work

Backbones for 3D object detection

- point-based
  - raw point clouds를 point operators로 direct하게 바로 feature추출
- range-based
  - range image들을 받아서 사용
- grid-based
  - point clouds를 voxel이나 pillar로 rasterize해서 3D network로 BEV feature map을 추출함
  - feature는 2D conv로 3D objects를 최종적으로 예측
- 최근 grid-based방식이 효율적이면서도 성능이 가장 좋음
  - Point2Seq는 대부분의 grid-based backbone과 함께 사용가능

3D objects prediction mechanism

- point-based
  - PointRCNN은 raw input의 key points’ location으로 바로 3D object proposal을 생성
- range-based
  - RangeDet은 range image의 pixel위에 3D bounding box를 추출
- grid-based
  - SECOND는 BEV map의 각 center에 다 anchor를 만들고, positive anchor에 대해서 object를 추출
  - SA-SSD는 part-sensitive warping으로 spatial feature를 향상
  - CenterPoints는 BEV pixel이 object ceenter 주변에 있으면 positive로 보고 bbox를 예측함
- 대부분의 방식들은 3D objects에 대한 attributes를 동시에 예측하고 intra-object의 information을 고려하지 않음
  - Point2Seq은 object내 attributes의 interdepency를 고려한 모델임

Set-to-Set matching for object detection

- DETR에서 처음 제안함
  - bipartite matching으로 object query가 각각 gt와 매칭되도록 함
  - 후속 연구들이 쭉 개선해옴
- 3D object detection에서도 사용되기 시작
  - 3DETR도 sampled points를 object query로 사용함
- Point2Seq는 sparse object query나 multi-step refinement가 필요하지 않음
  - BEV map에서 병렬적으로 decoding하고 similarity-based 방식으로 매칭해서 사용함
  - 추가적인 module이나 parameter가 필요없음

Detecting 3D objects as Sequences [Method]

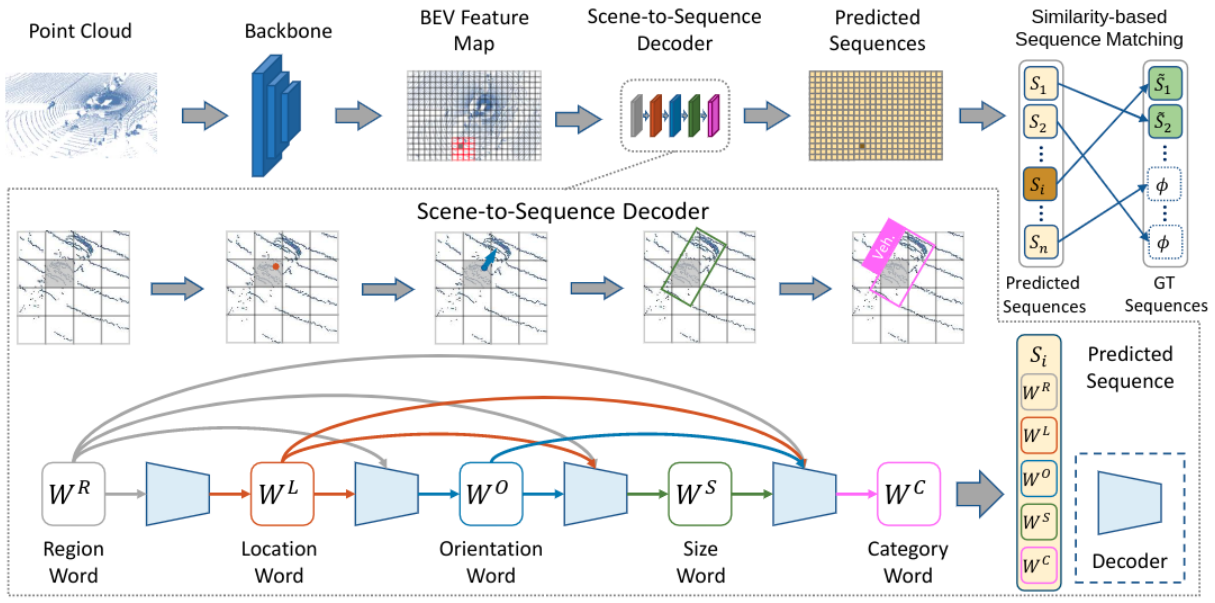

Figure 2. The overall architecture of our framework. Point2Seq contains three major components: the 3D backbone, the scene-to-sequence decoder, and the similarity-based sequence matching scheme. The 3D backbone takes the rasterized point cloud as input and outputs the Bird-Eye-View (BEV) feature map 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.

3.1. Architecture

- 3D bounding box를 예측, bbox개수*8
  - x,y,z/l,w,h/theta, class
    - x,y,z는 object의 center, l,w,h는 size, theta는 orientation, c는 class
    - W^R, W^L 이런 덩어리는 한번에 뽑는건지?
- 모델은 총 3가지 부분으로 구성되어있음
  1. 3D backbone
     - point cloud를 받아서 voxel형식으로 변환한 후, BEV feature map을 생성
     - BEV는 H*W*C로 구성됨
  2. Scene-to-sequence decoder
     - BEV feature와 initial set of region cue를 받아서 3D object에 대해 describe함
     - 뒤에 자세히 나옴
  3. Similarity-based sequence matching

3.2. Scene-to-Sequence Decoder

Problem formulation

- 이전에는 feature를 받고 object에 대한 attribute를 동시에 쭉 뽑는 방식으로 했음

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

  - 효율적이라 자주 사용해옴
  - object의 실제 영역과 BEV feature의 misalignment문제가 컸음
  - multi-stage refinement로 조금 개선했지만 연산량이 많음
  - sequential모델이 misalignment를 해결
    - **우에 해결??**

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.

Scene-to-sequence prediction

- 제안한 scene-to-sequence decoder는 BEV feature이랑 W^R들을 받아서 이후의 word들을 다 뽑는다.
- j step에서, W^R주변에서 spatial sampler S로 각 예측한 단어로부터 point들을 뽑음
- p의 hidden vector들을 aggregating함으로써 update를 함