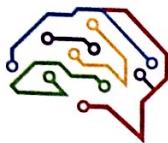




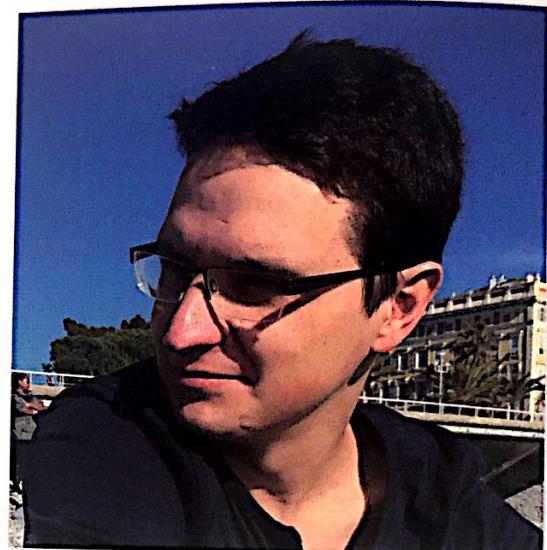
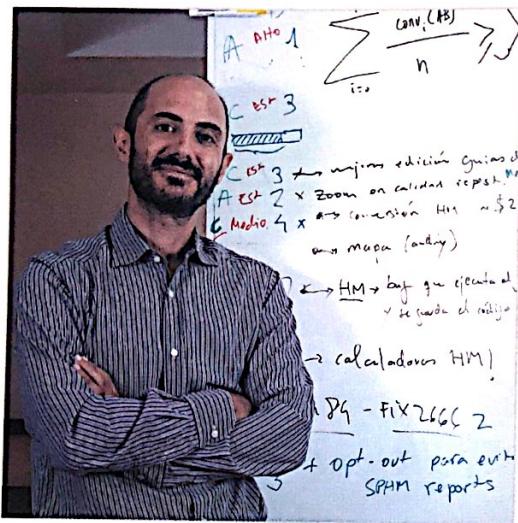
2018 CCF BDCI 6th
大数据与计算智能大赛
Big Data & Computational Intelligence Contest

大赛成果展示



2018 CCF BDCI 6th 大数据与计算智能大赛

Big Data & Computational Intelligence Contest



Pavel Gonchar:

Education: Belarusian State University (Faculty of Applied Mathematics and Computer Science)

Experience: Computer Vision Research (object recognition & detection, image manipulation, visual segmentation, neural networks on mobile development, performance optimization), Kaggle Master

Andres Torrubia:

Education: Polytechnic University of Valencia, Spain (Telecommunication engineering). Self-driving car nano-degree (Udacity/Online).

Experience: Researcher at CERN, cryptography/compression (10 US granted patents), serial entrepreneur (launched and sold startups in USA), Eisenhower Fellow, deep learning exposure via Kaggle competitions and #4 winner in DIDI/Udacity Self-driving car challenge (2017).

自动驾驶三维点云分割

自动驾驶三维点云分割 sanchinarro

Pavel Gonchar:

Education: Belarusian State University (Faculty of Applied Mathematics and Computer Science)

Pavel Gonchar:

Education: Belarusian State University (Faculty of Applied Mathematics and Computer Science)

Team Profile

Andres Torrubia is a Telecommunications engineer from the Polytechnic University of Valencia, Spain. Andres is the CEO and co-founder of Fixr, Inc an online marketplace for remodeling services operating in the US and Spain. Andres is the author of 10 US patents and is an Eisenhower Fellow. Andres studied deep learning with online resources: Stanford youtube classes, Udacity, arxiv, etc. and finished #4 in DiDi/Udacity self-driving challenge (competing against 2000+ teams) in August 2017.

Pavel Gonchar is senior machine learning engineer at Fyusion Inc (San Francisco, USA) with a background in computer vision research (object recognition & detection, image manipulation, visual segmentation, neural networks on mobile development, performance optimization). He is originally from Minsk, Belarus.

Andres and Pavel teamed up in September 2017 for online deep learning competitions, and have never met in person prior to this event.

Abstract

We present a novel transformation of the lidar point cloud dataset provided by Alibaba for the BDCI CCF 2018 competition into dense matrix that makes it straightforward to apply deep learning segmentation architectures typically used in image problems.

We explore different architectures, training regimes, regularization methods and test-time-augmentations and provide a real-time system that scores the winning IoU metric across all phases of the BDCI CCF 2018 competition within the real-time performance constraints (<100 msec).

Furthermore, we provide a method to trade-off score for computing power which operates in just ~25 msec with a very small impact in score.

Keywords

lidar, PCL, point cloud processing, segmentation, real-time, projections

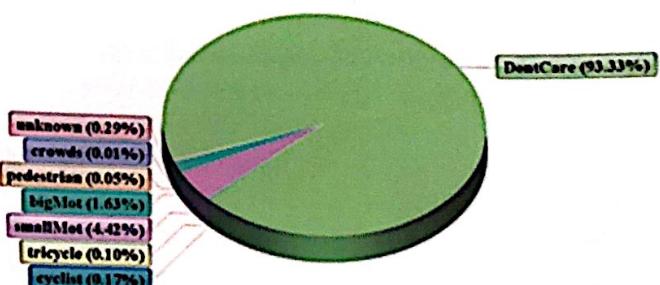


Fig. 1: Imbalanced data representation

There were two distinct test sets during the competition, one for the preliminary match with 13388 frames and other for the rematch part with 20086 frames. The test set format is the same as the training set but we are not provided with the labels.

The competition is scored by calculating the IoU of each category and then calculating the average IOU of the seven categories.

1.2. Related Work

A. Methods that directly consume 3D point cloud data

[1,2] are novel approaches that provide deep neural network architectures that directly ingest unordered point clouds as input and can work in segmentation or classification problems, among others. While they have the advantage of working directly on any type point cloud -including those generated by LIDAR sensors- we were not sure we could achieve high scores and still be under the 100 msec requirement of this competition, especially in light of the run-times reported in [5].

B. Point cloud projection methods

A different approach to this problem is given in [3]. This work demonstrated a 2D spherical grid projection of 3D cloud data to obtain a more compact representation (Fig. 2).

$$\theta = \arcsin \frac{z}{\sqrt{x^2 + y^2 + z^2}}, \bar{\theta} = [\theta / \Delta\theta],$$

$$\phi = \arcsin \frac{y}{\sqrt{x^2 + y^2}}, \bar{\phi} = [\phi / \Delta\phi].$$

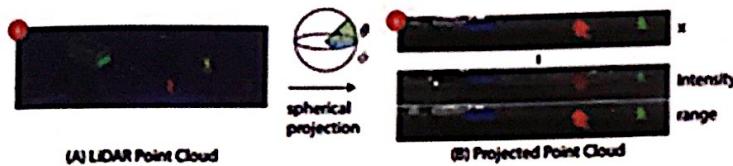


Fig. 2: LiDAR projections used in [3]

While this method reported very fast run-time performance, it leads to quantization artifacts and data loss due to the projection mapping: different 3d points may be projected to the same 2d point space.

C. Methods that combine both point cloud and RGB data

RGB images contain their own color information useful for segmentation. There could be a score boost from being able to combine multi-sensor information [4]. Unfortunately, we were not provided with RGB data in this competition so we did not pursue this further.

Proposed Solution

2.1 Data transformation

A lidar sensor produces a cloud of (x, y, z, i) values where x, y, z are spatial locations and i are values of reflected intensity by lidar beams.

Fig. 3 provides a rough sketch of a lidar sensor, consisting of multiple laser emitters and receivers mounted vertically on a rotating cylinder. Each beam fires periodically and its reflection is measured by a matching laser receiver. If the (x, y, z, i) point being measured is referenced against the receiver absolute position, given a particular beam the zenith \square angle should be constant, and we should be able to rearrange the measured points by azimuth \square angle to be either monotonically increasing or decreasing.

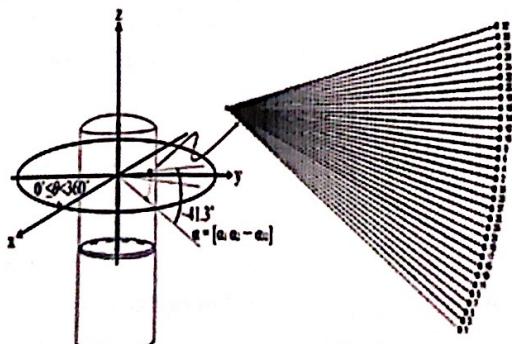


Fig. 3: sketch of a 32-beam lidar device

We performed intensive data analysis of the lidar dataset provided and determined the following:

The lidar device used to produce the dataset contained 31 usable beams (rings, hereinafter).

Rearranging the (x, y, z) points as (d, i, \square) as a $32 \times N$ matrix (calculating N so points in a frame remain the same) revealed that rings were fired sequentially and always in the same order.

Notwithstanding the above, the azimuth \square angle was misaligned with the pattern revealed by Fig 4.



Fig. 4: azimuth \square as lidar beam traverses the frame

The (d, i, \square) $32 \times N$ matrix can be shifted so that azimuth \square becomes almost aligned (error is within 0.1 degrees).

Once the above observations are realized, we can transform the input point cloud into the following channels:

- 1) $d = |xyz|$ (distance)
- 2) $i = \text{intensity}$ (same as originally provided)
- 3) $\square = \arcsin(x/y)$ (azimuth)

Fig 5 shows an example of visualizing $31 \times N$ matrix as 3 1-channel images.

Fig 5 shows an example of visualizing $31 \times N$ matrix as 3 1-channel images.



Fig. 5: dense matrix for d, i, \square viewed as images.

自动驾驶三维点云分割

Furthermore, we found that (x,y,z) values were referenced against a reference (0,0,0) at the temporal start of the frame. If we car where the measuring lidar moves, the resulting points are still referenced against this reference. We found that ring 0 (as per the 32xN matrix) actually contained the lidar location against that reference, and we can reframe the point cloud so that each (x,y,z) is relative to the location of the lidar device at measuring time, not at the beginning of the frame. This led to constant and discrete zenith Δ angles as we expected per Fig 3.

2.2. Deep learning model

We built a segmentation model using the popular U-Net architecture [6] as the foundational block with the following considerations:

Class-invariance in our matrix should be translational only in the X-axis (the Y-axis corresponds to rings and the same object captured by the lidar sensor changes shape as dimensions as it is further away from the sensor).

We can process a frame at once, or split a frame into overlapping regions and reassemble predictions.

Fig 6 shows an overview of our model, which has 9.8 million parameters. For regularization we added 0.1 dropout rate in all convolutional layers except the last one. We used Adam with an initial learning rate of 1e-2, reducing by factor 0.2 if validation loss did not improve in 3 epochs, batch size was 16. We used a total of 2x1080 Ti, 2x2080 Ti and 4x1080 Ti GPUs during all our multiple experiments. We trained our best model for 105 epochs.

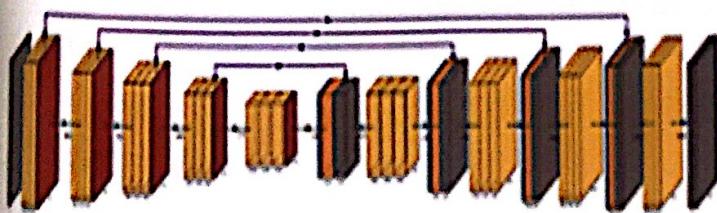


Fig. 6:segmentation model

We used a mixed loss function consisting of the Tversky loss [7] with parameters alpha=beta=1, plus alpha=beta=0.5 scaled by 0.5 and categorical cross-entropy. We also tried the Lovász-softmax loss [8] but it did not improve score.

2.3. Score and performance

Our model showed consistent performance across all stages of the competition:

Preliminary match A: 0.31379882 \times Preliminary match B: 0.31198952 \times Rematch match A: 0.32195514 \times Rematch match B: 0.31749678

It takes ~65 msecs on a 1080 Ti graphics card using 4 overlaps and a 31x1808 model. We trained a model that can accommodate the biggest frame (31x1818) in a single pass and it achieves comparable (barely less) performance in just ~25 msecs.

Further steps

We suggest our dense transformation can be used by LIDAR manufacturers to provide point cloud data in a deep-learning friendly format.

References

- [1] Qi, Charles R., et al. "Pointnet: Deep learning on point sets for 3d classification and segmentation." Proc. Computer Vision and Pattern Recognition (CVPR), IEEE 1.2 (2017): 4.
- [2] Li Yangyan et al 2018 PointCNN arXiv preprint arXiv: 1801.07791
- [3] Wu, Bichen, et al. "Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud." 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018.
- [4] Chen, Xiaozhi, et al. "Multi-view 3d object detection network for autonomous driving." IEEE CVPR. Vol. 1. No. 2. 2017.
- [5] KITTI Object Detection Evaluation 2012 leaderboard results, http://www.cvlibs.net/datasets/kitti/eval_object.php
- [6] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox.

自动驾驶三维点云分割

自动驾驶三维点云分割

杨超越粉丝团

俞依杰

车辆工程&硕士研究生二年级

江苏大学

中国-镇江

13588280094@163.com

娄新雨

车辆工程&硕士研究生二年级

江苏大学

中国-镇江

louxy126@126.com

费敬敬

计算机&硕士研究生二年级

同济大学

中国-上海

feijingjing1994@foxmail.com

摘要

本算法首先剔除3维点云中处于所选择的兴趣区域外的点，并将剩余点投影为2维栅格地图。之后，以每个栅格中点的归一化密度作为特征组成密度图、以每个栅格中点的最大高度作为特征组成高度图、以每个栅格中点的最大反射强度为特征组成强度图特征组成强度图，并分别以上述三种图为R、G、B通道，组成一幅RGB图像，作为检测端的输入。然后使用基于图像的深度学习目标检测算法ROI-align的faster-rcnn+fpn对图像上的目标进行检测；最后根据检测结果对点云进行类别标记。在i7 CPU+GTX 1080 GPU的硬件上达到至少10帧每秒的处理速度，满足赛题要求。

关键词

点云, RGB图像, faster-rcnn+fpn

俞依杰, 娄新雨和费敬敬。2018. 自动驾驶三维点云分割。在CCF大数据与计算智能大赛决赛答辩论文集中。BDCI, 浑南区, 沈阳市, 中国, 3页。

1赛题的理解

本赛题要求将点云分为自行车,三轮车,小车,大车,行人,人群,未知障碍物(在可行驶区域但难以识别为前面六个类别的物体)和背景(除去前面七个类别的其他所有点)八个类别,并要求参赛者的方案在i7 CPU+GTX 1080 GPU显卡的硬件上达到至少10帧每秒的处理速度。相比较Kitti等数据集仅将障碍物分为车辆、行人、非机动车,并对运行环境无要求,难度大大增加。

目前主流的点云数据处理方法有两类。一是直接基于点云的处理方式;另一种是基于栅格地图的方式。由于目前开源的直接处理点云的算法所针对的多为室内场景,对本赛题的复杂室外场景效果较差,同时考虑到实时性的要求,我们选择简单稳定的2维栅格地图作为我们的处理对象,并结合2维图像上的深度学习检测算法进行目标识别。即将点云分类问题转化为图像目标识别问题。

2数据探索

由于基于图像的目标检测算法所输入的为RGB图,因此传统的二值栅格地图并不适用于这些算法,在此,我们借鉴了complex-yolo的思路。在complex-yolo中,作者首先将点云转化为鸟瞰图形式的栅格地图,并将每个栅格的归一化密度、最大高度、最大反射强度

作为RGB图像的三个通道,并使用简化的YOLOv2体系结构,通过复杂的角度回归和E-RPN进行扩展,以在实时运行的同时,实现检测准确的多类别和方向的3D物体。由于赛题所给数据为自主车周边360°的环境数据,车辆侧面和后方的障碍物同样需要检测。因此,我们将以雷达为中心、长Height、高Width的矩形区域作为感兴趣区域(ROI),并将该区域的点云制作成RGB图,感兴趣区域之外的点直接分类为背景点。RGB三个通道的特征选择与complex-yolo相同。其中,归一化密度r计算式为:

$$r = \min \left(1, \log \frac{(N + 1)}{64} \right)$$

其中,N为栅格中点的数量。

3方法尝试

3.1数据预处理

由于本次赛题对实时性有较高要求,因此我们在ROI的尺寸即Height、Width的选择上进行了多次尝试,使得算法能够兼顾实时性和检测准确性。

同时,我们还尝试了在激光雷达点云处理方面使用较多的最大最小栅格地图。最大最小栅格地图:统计了每个栅格中所有点的最大高度与最小高度,得到了两者的相对高度差。当高度差大于阈值时,则认为此栅格是占据的,即存在障碍物。该方法能够滤除非障碍物点尤其是地面点。但是经过尝试之后,我们发现监测的准确率下降较大,因此放弃该种方法。

RGB图像的值得范围为0-255的整数或0-1的浮点数。在初赛阶段,我们将计算得到的归一化密度、最大高度、最大反射强度等比例扩展至0-255范围内的整数,所得图像如图1(左)所示。在复赛阶段,我们直接将计算结果所得的浮点数作为值输入,所得图像如图1(右)所示。经过验证,在检测端使用同一网络的情况下,方法二检测准确率较高,因此我们选择该方法进行图像生成。

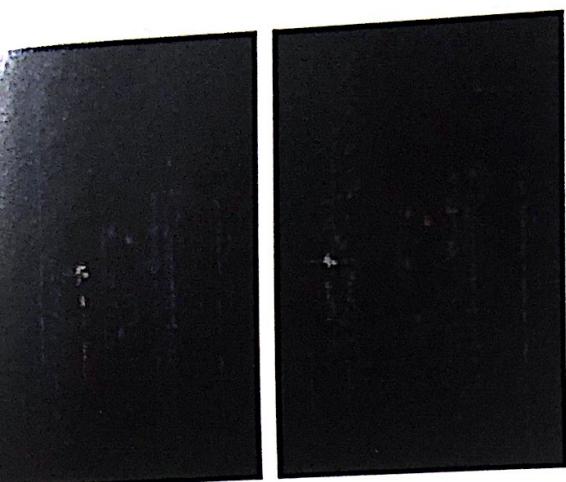


图1：(左)方法一生成的图像 (右)方法二生成的图像

3.2 检测端

在初赛阶段，我们是直接参考complexyolo文中的方法，使用原始的yolov2作为目标检测算法，得到了一个0.008的基线成绩，在用yolov3作为替代后，得到0.009的一个基线成绩，这两个单阶段检测算法虽然检测速度较快，但是检测精度始终不尽如人意，我们决定采用一些检测速度慢，但精度更高的目标检测算法。我们首先尝试了目前检测效果最好的单阶段模型RetinaNet作为检测算法，使用的特征提取网络是resnet101，在未调参的情况下就得到了0.1672的基线成绩，远胜于之前的yolo系列算法。与此同时，我们也尝试了一下简单的模型融合方法，我们将yolov3和Retinanet的检测结果进行了加权平均，但并没有得到更好的成绩。最终以初赛B榜排名第8的成绩进入了复赛。

在复赛阶段，一开始我们仍旧试图针对Retinanet进行调参来取得更好的成绩。首先，我们尝试了若干种不同的特征提取网络，由于实时性要求的限制，我们不能使用较深的特征提取网络，在尝试了densenet，inceptionnet后成绩都没有得到较大的提升。其次，我们考虑到假阳性可能会对得分造成很大的负面影响，对检测框置信度阈值进行了限制，经尝试由原来的0.5调整为0.55，使得成绩有一定的提升，但提升不大。然后，在观察了检测结果后，发现有大量的检测框相交在一起，却没有被nms算法滤除，我们降低了nms算法的iou阈值，也使得成绩有一定的提升。最后通过上述方法，我们将RetinaNet的成绩提升到了0.178左右，总的来说仍旧没有大幅度提高。因此，我们开始考虑使用当前检测效果最强的目标检测网络之一的Faster-rcnn+fpn。由于Faster-rcnn+fpn，下面简称为Faster-fpn，网络参数较多，不论是推断阶段还是训练阶段都要花费更多的时间，为了满足实时性的要求，我们只能把点云图的范围缩小，呈现在目标检测端的影响就是，输入的图片尺寸缩小为了900×600。在没有调整任何参数的情况下，我们取得了0.244的A榜成绩，但我们同时也发现了一个问题，就是我们的检测算法并不是端到端的，一开始采用的一边生成图像高维数组，一边进行检测

的思路，是不符合组委会所说的进来一帧检测一帧的要求。而900×600的检测图若是采用串行的运作方式，仍旧无法满足实时性要求，在多次尝试后，我们发现700×500的图片的推断过程几乎能够保证10fps的实时性要求（会有部分图片超过0.1秒，但是大部分都在0.08至0.09之间）。在确定了700×500的检测图思路后，我们开始针对网络参数进行优化。总体思路与Retinanet的思路相似，1.基础特征提取网络最后我们选择的是resnet101；2.降低nms阈值后，发现得分反而更低；3.调整置信度阈值，首先我们是把所有类别的置信度阈值都设为0.5，滤除所有低于该阈值得分的目标，但发现得分也没有提高，而由于我们使用了更小的检测图，又减少了检测范围，反而使得我们的得分相较之前降低了很多，仅仅只有0.20左右的得分。考虑到每天的提交只有3次，不能胡乱的提交，而验证集上调参后得分的差距又不明显。我们决定进行人工验证，我们用模型去检测划分得到的验证集，挑选了30张明显检测效果不好的图片，记录下文件名称后，将这些验证集的groundtruth也投影到俯视图上，与检测结果进行对比。我们发现自行车，unknown对象是误检的重灾区。大量的背景点被检测为了自行车，unknown，以及这两类相互之间的误检。而车辆点的会因为置信度的调高而出现漏检，说明训练样本的种类不均衡比较严重，某些类别的训练样本较少，质量较差，造成假阳大大影响了检测得分。因此我们不再统一地对所有类别进行置信度限制，仅仅针对某些类别进行置信度限制，经过几天的实验，我们最终选择了对自行车施加0.6的置信度阈值限制，对行人施加0.4的置信度限制，对unknown设置0.3的置信度限制，最终得到了接近900乘600基线的得分。部分调参结果如图2(左)、(右)所示。

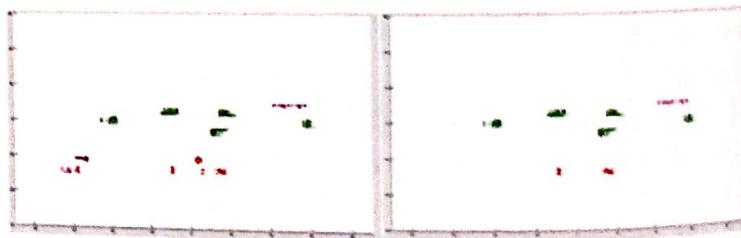


图2：左：unknow（紫色）置信度为0.4，自行车（红色）置信度为0.6的检测结果；右：unknow（紫色）置信度为0.6，自行车（红色）置信度为0.8的检测结果

4 模型优点

本模型最大的优点是，精确度相较直接使用点云作为输入的方法会有较大的提高，而且由于精度手目标检测算法的影响较大，同时又能轻松地移植到不同的目标检测算法上去，如今目标检测算法精度不断提高使得本模型具备了精度不断提升的潜力。同时由于同时融合了点云数据和传统图像检测算法，使得本模型可以应用于多种领域，比如为传感器融合算法提供了一种解决思路，用于模型融合提高普通RGB图像目标检测算法的精度等。

自动驾驶三维点云分割

5后续可以继续提高的点

首先，我们团队没能很好地进行数据分析处理，意识到训练集中种类数量偏差过大时已经为时已晚，若后续能够对训练数据做好仔仔细地数据清洗工作，可能能够得到更好的成绩。

其次，我们没有尝试最新地目标检测算法，如CornerNet等算法，也没有大幅度地针对此次任务改动过网络结构，对卷积层进行调整等，仅仅是做了简单地调参工作。

最后，我们选用地目标检测算法是来自于使用tensorflow框架的tensorpack库，没有使用基于caffe2的实时性更好的detectron库的图标检测代码。

致谢

感谢教育部高等学校计算机类专业教学指导委员会、沈阳市人民

政府、中国计算机学会等组织本次比赛的单位，以及竞赛平台

DataFountain。感谢王海老师对于本次比赛的指导。

自动驾驶三维点云分割

自动驾驶三维点云分割 lidar_debug

曾钰廷
算法工程师
长沙智能驾驶研究院
中国-长沙
nhzyt@163.com

团队简介

个人参赛，主要从事深度学习方面的工作，包括图像和激光雷达领域相关工作，熟悉PCL, OpenCV, Caffe, Tensorflow等工具。

2018年7月硕士毕业，毕业后开始从事激光雷达方面的工作，主要利用lidar进行障碍物检测和分类。

2017年，曾在三星北京研究院实习，研发基于移动端的手势识别等深度学习和图像方面的工作。

2016年，曾在清华大学计算机系实习，参与深度学习相关的目标检测及人脸识别方面的工作。

项目经历：

1:FPGA神经网络芯片：利用FPGA加速深度学习模型。

2:人脸识别和目标检测：安防监控环境下的目标检测和人类识别。

3:图像检索：利用卷积神经网络开发基于哈希的服装检索系统。

4:激光雷达目标检测：自动驾驶领域的lidar相关的点云分割和目标检测。

5.lidar到camera标定：传感器之间的标定工作，建立一个坐标系。

摘要

自动驾驶技术受到越来越多的关注，而激光雷达传感器几乎成了自动驾驶车辆的标配，三维点云语义分割就是其中的热点问题。

考虑到自动驾驶系统的计算能力有限，我们采用三维点云栅格化的方法，将三维点云数据降维到二维统计信息。同时，为了能够更好的表达三维点云信息的特征，借助卷积神经网络，这一在图像领域展现强大特征表达能力的网络。

由于数据集的限制，面对32线的激光雷达设备，采用了俯视图的栅格化[1]方法，而不是前视图栅格化[2]的方法。为了加快计算，将三维点云投影到俯视图中的栅格内，并统计栅格内相关的统计量。这样可以得到类似于图像RGB的相关统计量，从而可以借助卷积神经网络强大的特征提取能力。将二维的相关统计量送入到分割模型中，从而可以利用FCN[3], PSPNet[4]等分割网络，利用卷积运算和反卷积运算，从而可以得到栅格内的点云类别以及所属类别的置信度。由于数据类别的不平衡，不能直接利用softmaxloss等损失函数训练，必须对不同类别进行不同的权重，所以可以统计数据集中各类别的比例，进行加权计算。

关键词

三维点云，栅格化，卷积神经网络

BDCI参考格式：

曾钰廷.2018. 三维点云分割. 在CCF大数据与计算智能大赛决赛答辩论文集中。BDCI, 浑南区, 沈阳市, 中国, N页。

1 点云俯视图栅格化

针对32线的lidar数据，每帧产生6万个点，相对数据量还是较大，为了能够更快的运算，所以进行俯视图栅格化处理。将三维(x,y,z)点云坐标，投影到创建的(x,y)网格中，并且对空间中每个点云分配到一个网格中，那么位于网格中的点可以统计出多个特征量，例如位于网格中点的最大高度。可以得到类似于图像RGB形式的数据结构，那么就可以借助图像中的语义分割网络，将其应用到激光雷达领域。

而且，栅格化处理几乎不耗时，能够快速的将三维点云信息降维二维的信息。因此，如图1所示，可以统计每个栅格内的最小高度值，最大高度值，最大反射强度，平均反射强度，点云个数等栅格内的点云统计信息。

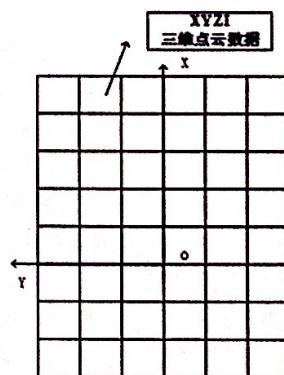


图1:三维点云栅格化示意图

2 点云分割网络

栅格化处理可以得到初步的点云特征，但是还无法描述真实的目标的特征。因此，借助于卷积神经网络进行深层的特征提取。于是，输入分割网络的数据格式为[height, width, channel]，其中height和width可以根据不同lidar的探测距离以及点云的水平和垂直分辨率来决定。例如，32线lidar，栅格大小可以设置为0.12m，探测距离为60m，那么height= 60/0.15=500。channel则是栅格内统计量的个数。

2 点云分割网络

栅格化处理可以得到初步的点云特征，但是还无法描述真实的目标的特征。因此，借助于卷积神经网络进行深层的特征提取。于是，输入分割网络的数据格式为[height, width, channel]，其中height和width可以根据不同lidar的探测距离以及点云的水平和垂直分辨率来决定。例如，32线lidar，栅格大小可以设置为0.12m，探测距离为60m，那么height=60/0.15=500。channel则是栅格内统计量的个数。

4 模型训练细节

三维点云进行栅格化后，那么对应的分割模型的label也就是所在栅格对应的label。因此，需要根据数据集点云的label重新计算栅格所对应的label，对于没有点云数据的栅格则设置为数值8，不参与loss计算。由于栅格化后，栅格内点云的类别可能不只一个，因此统计点云个数最多的label为栅格label，数值为0到7。如图3(a)白色区域没有点云数据则不进行损失函数的计算。因此，剩余的部分，绝大多数位置是小车和背景的点云，如图3(b)是相对(a)中场景的轿车在lidar场景中的分布情况。

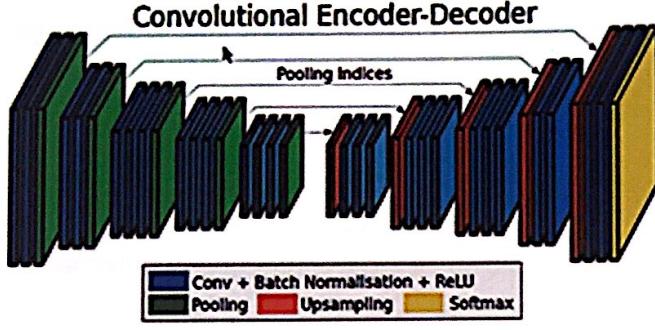


图2：语义分割网络示意图

3 损失函数设计

将三维点云栅格化后，那么分割网络针对的就是类似图像中像素级别的损失函数的设计。由于点云很稀疏存在大量非点云的栅格，同时这些栅格也不存在对应的点云label，那么将不进行损失函数的计算。除去非点云的栅格，还是存在大量的背景点云的栅格，较多的轿车点云的栅格以及其他类别点云的栅格，单帧点云数据中类别不平衡以及数据集中类别不平衡性非常严重。因此，必须针对类别的不平衡性进行损失函数的优化。

如公式1为softmax loss的损失计算公式，针对lidar的损失明显不适用，因此可以对不同类别的数据进行加权计算，加权值可以根据栅格所对应类别的数量决定。使得类别少的数据获取较大的损失函数。如公式(2)，针对不同类别设置不同的weighting，可以通过遍历数据集中类别的个数，获取对应的数值，例如取倒数。

$$loss = -\sum_i^n y_i * \log f(z_i) = -\log f(z_k) \quad (1)$$

$$loss = -\log f(z_k) * weighting \quad (2)$$



图3：俯视图及小车label示意图

參考

[1] Luca Caltagirone, Samuel Scheidegger, Lennart Svensson Fast LiDAR-based Road Detection Using Fully Convolutional Neural Networks

[2] Bichen Wu, Xuanyu Zhou, Sicheng Zhao. SqueezeSegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud

[3] J. Long, E. Shelhamer and T. Darrell. Fully convolutional networks for semantic segmentation. In CVPR, 2015.

[4] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In CVPR, 2017.

自动驾驶三维点云分割

自动驾驶三维点云分割

SDU-IRC

冷汉超
SDU-IRC/2018级硕士
山东大学
中国-青岛
lh3538@gmail.com

曹金明
SDU-IRC/专业&年级
山东大学
中国-青岛
jinming.ccao@gmail.com

魏卓
SDU-IRC/专业&年级
山东大学
中国-青岛
weizhuosdu@gmail.com

王业超
SDU-IRC/专业&年级
山东大学
中国-青岛
wycsdu@gmail.com

潘志一
SDU-IRC/专业&年级
山东大学
中国-青岛
panzhiyi_96@outlook.com

团队简介

冷汉超,山东大学计算机学院交叉研究中心2018级研究生,本科期间曾获第五届“中国软件杯”大学生软件设计大赛国赛二等奖等奖项,大四于小米物联网事业部智能摄像头项目组实习,参与研发了米家智能摄像头PC端、目标检测等功能,目前主要研究方向为计算机视觉。

曹金明,山东大学计算机学院交叉研究中心2018级博士研究生,研究方向是计算机视觉。在比赛中负责PointCNN强度信息的优化,之前主要研究深度神经网络在图像中的应用,目前主要研究3D视觉的应用。

魏卓,山东大学计算机科学与技术学院研究生,主要研究方向为计算机视觉。曾获得陕西省ACM程序设计竞赛金牌。

王业超,本科就读于山东大学生物信息专业,现为2018级研究生,就读于山东大学计算机学院交叉研究中心,从事计算机视觉研究。主要研究方向为3D点云感知。

潘志一,山东大学计算机科学与技术学院2018级研究生,方向计算机视觉。

关键词

自动驾驶,三维点云分割,PointCNN,点云预处理

1赛题的理解

传统方法对于点云分割问题的解决一般依赖于树、图等经典结构,通过一定的特征提取方法来对目标点云进行分割。这样往往面临很大的问题。首先,传统方法的特征提取和参数选择很大程度上依赖于经验;其次,对于噪声、旋转、缺失的情况,传统方法不能很好的处理;另外,很多传统方法依赖大量迭代,导致算法时间复杂度高,随着点云规模的增长,其运行速度增加较大,难以满足需求。

与之相对的,近年来被广泛采用的深度学习方法可以“学习”到网络参数,不需要人工进行复杂的特征工程;对于点云的缺失,旋转等情况也具有一定的抗性;另外,尽管深度学习需要较长的训练过程,但是在网络训练完成之后,单次前向传播较快,能在测试时达到较高帧率。

基于我们对于数据的分析,目前在各种领域表现优异的深度学习方法无疑是本次比赛中最有力的候选项。

2方法选择

由于本赛事数据与深度学习的方法契合度较高,因而我们在设选定方案时主要考虑基于深度学习的方法,首先调研近期针对点云分割任务表现突出的几种方法:PointNet[1]、PointCNN[2]、PointSIFT[3]、PointSeg[4]等,使用比赛数据集在各个方法上训练测试,建立Baseline,从中选择出速度较快和IOU较高的作为继续的基本方法。

在确定基本方法前,我们对前期调研的几种方法进行了对比,使用相同数据集和默认分割任务的参数训练。

IOU:PointSIFT > PointCNN > PointSeg > PointNet

摘要

在过去的这些年里,对于二维图像已经有了大量深入的研究,而对于3D相关的研究还有很大提升空间,本工作借助此次比赛开放的面向自动驾驶的三维点云数据,探索针对自动驾驶的高效三维点云分割方案。在本次比赛中,我们测试了主流点云分割的解决方案,包括PointNet、PointCNN、PointSIFT、PointSeg,综合考量各个方案在速度和准确率上的表现,并最终确定在PointCNN的基础之上进行改进。改进过程中,采用点云分块的预处理方法,解决了直接在整帧点云中采样过于稀疏的问题;采用按类别数量比例调整loss weights的方法,解决了数据集类别数量失衡的问题。最后,列举了本工作继续改进的途径和方法。

速度: PointSeg > PointCNN > PointNet > PointSIFT
 经过实验发现PointCNN在速度和IOU上都表现较好,最终确定在PointCNN框架基础上进行改进。

3数据探索

本次数据数量庞大(共50000帧点云),点数充足(平均每帧50000个点),规模较大(百米以上范围),种类多样(7类),对于每个样本点都具有标注信息,采用较为主流的三十二线雷达扫描结果,这为使用深度学习方法创造了基础。
 数据预处理前,我们统计了训练集中各个类别中点的数目,发现背景点特别多,而行人的点极少,数据集类别数量严重失衡,会导致行人的预测结果很差。

4数据预处理

由于网络结构需要固定的输入,所以点云在输入网络前,先会从整个场景(约50000个点)中采样固定数量的点(测试时使用2048个点)作为输入,但是从整个场景中采样一部分点会让输入变得非常稀疏,丢失大量空间信息。
 所以我们尝试将一帧点云数据分成4块,每块约12500个点,从块中采样2048个点作为输入,这样就使得输入的点稠密了许多,空间信息自然也就更充足。
 具体分块策略是将一帧数据拆分成以坐标轴正负方向为中心线的四个90度视锥,每个视锥中所有点作为一个块,并将所有视锥的方向统一旋转到x轴正方向。

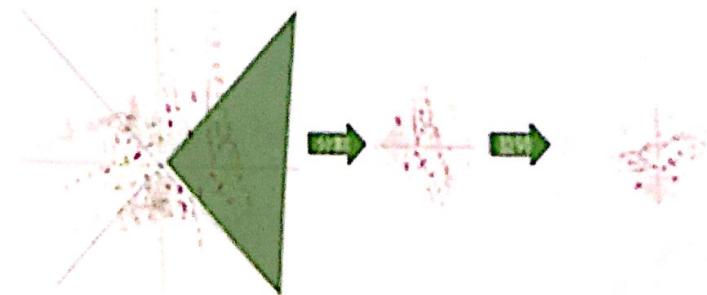


图1 数据分块过程

预处理前IOU得分为0.179,预处理后得分提升至0.239。

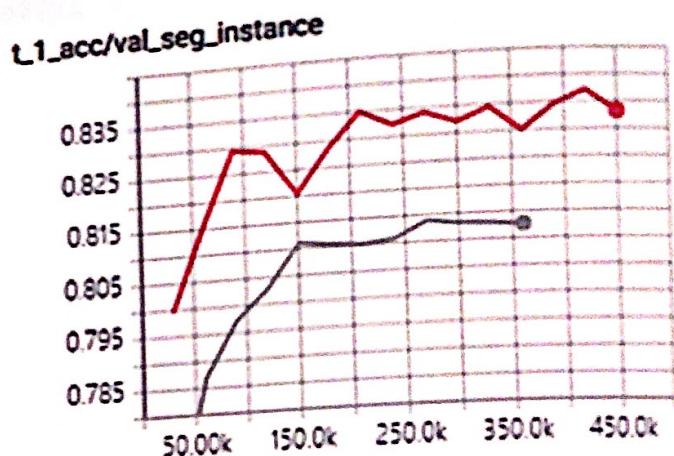


图2 数据预处理前(灰色)后(红色)的准确率曲线

5模型调参

基准模型训练参数采用PointCNN在知名点云分割数据集Semantic3D上使用的参数。以基准模型结果作为参考,微调卷积层kernel size、stride等参数,发现结果并没有提升。
 随后统计了比赛数据集中各种类点的数目,根据各类别点的比例设置模型的labels weights, IOU从0.146提高到了0.179。

t_1_acc/val_seg_instance

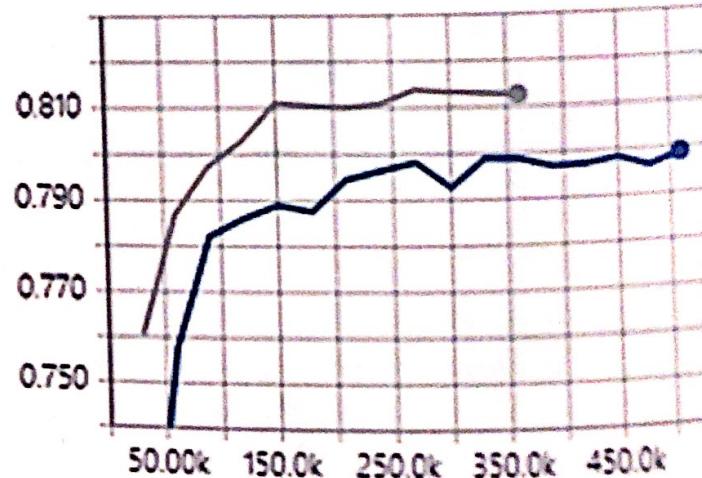


图2 labels weights调整前(蓝色)后(灰色)准确率曲线

此次比赛数据集规模是Semantic3D的3倍，所以适当增加迭代轮数和学习率衰减步数，IOU也有小幅提升。

在赛题讲解直播中得知数据集有近一万个点强度太弱，并且存在于原点附近，所以在数据预处理过程中将这些点去掉，预测时直接将这些点置为背景点，通过网络预测的点数减少，速度有所提升。

6后续提高点

这次比赛速度上并未达到极致，有很多需要优化的地方，例如数据分块过程的并行化、KNN的优化等。

致谢

感谢中国计算机学会以及沈阳市人民政府组织这次比赛，让我们有机会与各地同行同台竞技，感谢阿里巴巴开放的珍贵的三维点云标注数据，让我们有机会打开三维视觉世界的大门。

参考

- [1] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. arXiv preprint arXiv:1612.00593, 2016.
- [2] Y. Li, R. Bu, M. Sun, and B. Chen. PointCNN. ArXiv e-prints, January 2018.
- [3] M. Jiang, Y. Wu, and C. Lu. (Jul. 2018). “PointSIFT: A SIFT-like network module for 3D point cloud semantic segmentation.” [Online]. Available: <https://arxiv.org/abs/1807.00652>.
- [4] Y. Wang, T. Shi, P. Yun, L. Tai, and M. Liu, “Pointseg: Real-time semantic segmentation based on 3d lidar point cloud,” arXiv preprint arXiv:1807.06288, 2018.