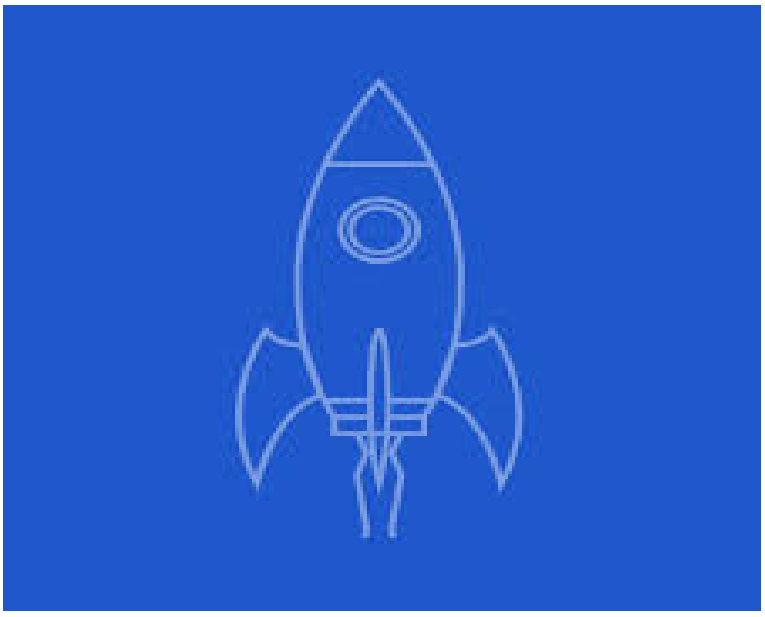

University of Colorado at Boulder

APPM 5720

Applied Deep Learning 1: Computer Vision

Sparse Recurrent CNNs for 3D Point Cloud Semantic Segmentation

---

Authors:
Cooper Simpson

Affiliations:
M.S. Applied Mathematics

December 9, 2020

Contents

1 Abstract

Effective and efficient understanding of 3D scenes (encoded in point clouds) is a key component of autonomous systems. Often, the 3D data being produced is sequential in nature (e.g. self driving cars), and because such systems need to operate in real time, the data also needs to be processed sequentially. In this work I propose a sparse convolutional neural network equipped with a recurrent module that can leverage this temporal dependency. A state-of-the-art sparse point-voxel convolutional network found using neural architecture search acts as a backbone to extract point cloud features. This data is then combined with hidden state memory – accumulated from previous frames – in the recurrent module. I propose a number of variations on this fundamental idea in terms of how data fusion occurs. I attempt to evaluate my model using the nuScenes semantically labeled LiDAR dataset for autonomous vehicles. We see that ... ¡Insert some more here¿

2 Introduction

There are a number of different tasks associated with the processing of 3D data in deep neural networks. For example, one may consider a single object or an entire scene of objects. The goal may be classification, object detection, or semantic segmentation. All of these tasks consume essentially the same data: 3D (e.g. x,y,z) locations in space. Further attributes may come in the form of color, LiDAR return intensity, object normals, and more. In many of the use cases for 3D deep neural networks (DNNs) accuracy is of the utmost importance. The canonical example of this is self-driving cars, but many tasks involving autonomous robots put others at risk. However, the overarching goal is further complicated by the fact that not only must we have accurate models, but they must be quick and efficient as well.

In this work I focus on the task of semantically segmenting sequential scenes of 3D LiDAR point clouds. To a certain degree this just boils down to individual point-wise classification. However, as compared to 2D images it is a much more difficult proposition given the scale of the data and its non-homogeneous sparse structure. A number of recent developments have made the problem significantly more approachable and tractable. Specifically, I identify the recent release of a number of sequential point cloud datasets, advances in sparse 3D convolutions, and the successful application of recurrent modules as allowing for my work.

For some time there have existed many publicly available datasets for the purpose of benchmarking and advancing 3D processing tasks. Most of these datasets only include smaller scenes (or single objects) and are not sequential in nature. Furthermore, when it comes to semantic segmentation the amount of work involved in annotating data is enormous given the large number of points in a 3D scene. The Waymo Open dataset, released in 2019, has 1,000 annotated sequences with a

rich diversity of sensors but lacks semantic labels. It is still important data and incredibly useful for object detection. The popular SemanticKITTI dataset uses the well-known KITTI Odometry benchmark and semantically labels each sequence. Following this, and competing with Waymo, the nuScenes dataset was released in 2019. They also include 1,000 sequences but have since updated their release with semantic labels for the LiDAR data. These datasets not only provide the correct type of annotated data, but they do so at a scale convenient for training large DNNs.

A common approach that is taken (indeed the route we choose here) when working with point cloud data is to voxelize. Voxels are the 3D equivalent of pixels, and the process of voxelization essentially turns a continuous cloud into a discretized grid where individual voxels may contain one, none, or many points. Importantly these voxel grids are extremely sparse because the point cloud itself is very sparse. As well, larger voxels will reduce the sparsity but will also eliminate fidelity necessary for inference. When it comes to these sparse structures traditional convolutions are very inefficient, thus the idea of sparse convolutions seeks to take advantage of the sparse nature to speed up operations without loss in accuracy. Much work has been done here on the theory and practical implementation. Suffice it to say that sparse convolutions greatly increase the compute capabilities of models operating on point cloud data.

It may seem as if applying recurrent networks to sequences of point cloud data follows naturally from their temporal structure, but the idea is relatively unexplored. Likely a number of factors have had an effect on this including the absence of the advances discussed above. As well, when it comes to DNNs recurrent networks are often much slower than something like a standard CNN as many of the operations cannot be parallelized.

My work seeks to combine the power of large annotated datasets, the efficiency of sparse convolutions, and the leverage of temporally dependent recurrent neural networks.

3 Related Work

There exists a plethora of research that informs this work directly and indirectly. I will not discuss all of this in detail, and instead I will focus on two pieces of work that have heavily inspired and facilitated the model I present here.

3.1 Sparse Convolutions

[7] describes a new operation: the Sparse Point-Voxel convolution. This equips a standard sparse convolution with another branch that applies an MLP to the points directly. The goal is to perserve some of the fidelity lost in the voxelization process. The authors present a library for conducting these sparse operations and also build a neural architechture search method to find an optimal model that uses these methods. They achieve state-of-the-art accuracy. A sparse

convolution in general only acts on the active sites (i.e. the non-sparse regions).

3.2 Temporal Based Detection

[4] describes their methods to use general sparse convolutions with an LSTM for object detection. A sparse convolutional U-Net is used as a feature extractor and also within the LSTM module. They also achieve state-of-the-art performance.

4 Experimental Setup

As stated before, the goal of the work presented here is to semantically label a sequence of LiDAR point clouds. Given a point cloud as input the model I propose uses an SPVNAS model found in [7] as a backbone feature extractor. These features are then passed to a recurrent module which fuses them with features from the hidden state (informed by previous inputs). This fused data is then used to classify each point as belonging to one of a preset number of classes. I will note here that given the training data is outdoor driving scenes, the model is only expected to preform on similar data.

4.1 Data Analysis

As our training and validation data we use the autonomous vehicle nuScenes dataset. This provides a large amount of training and validation data while still having semantic labels. There are 1,000 scenes of about 20 second length with keyframes annotated at 2Hz. This results in ¡insert amount¿. Unfortunately, nuScenes only uses one LiDAR sensor (as compared to the 5 of Waymo) which reduces the density of the point cloud. The provided data split consists of 700 training sequences, 150 validation sequences, and 150 testing sequences which do not have available annotations. As well, a miniature version of the dataset is available (with 10 sequences) which we have used for experimentation. A sample of a scene from this mini set can be seen below.

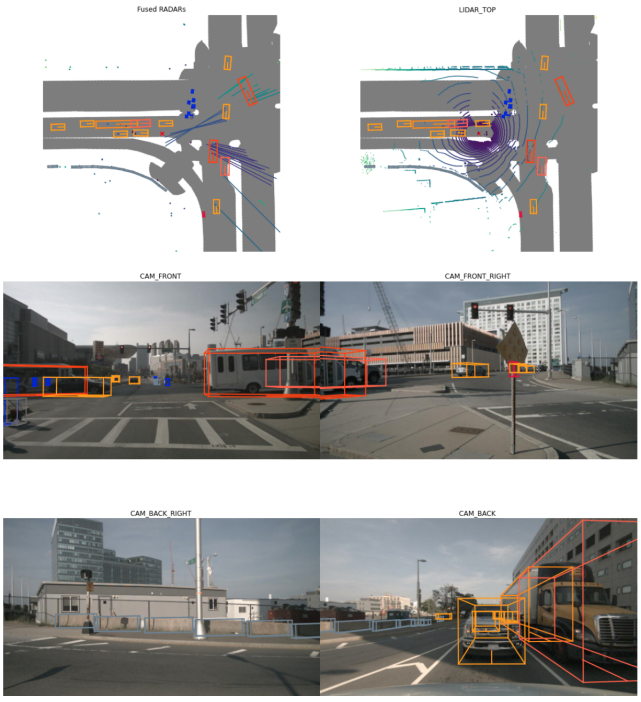

Figure 1: A view of a number of sensor outputs from a single scene.

We can also see another viewpoint of the same scene below, visualized using the *Open3D* software.

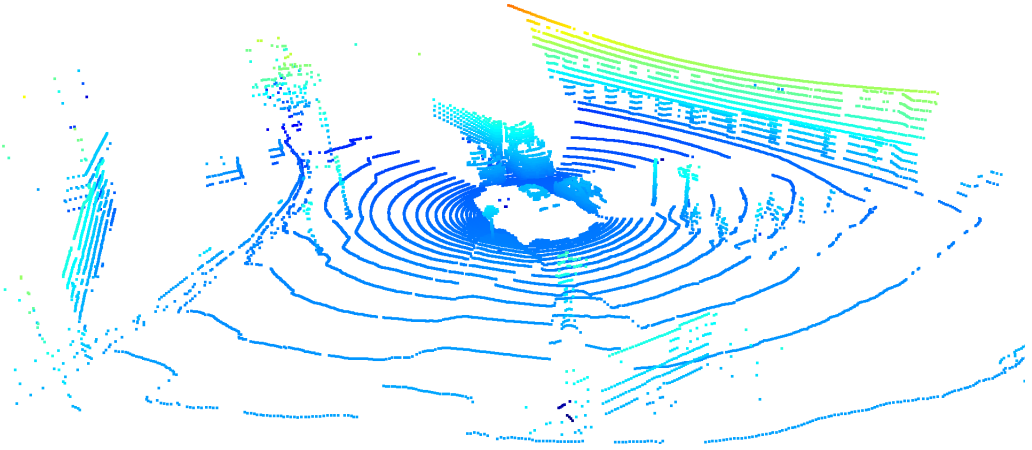

Figure 2: A view of the 3D point cloud of the same scene.

In our processing an input point cloud is voxelized using the *torchsparse* package according to the procces discussed in the previous section. We take our voxel edge length to be 0.05m. During

voxelization if a voxel contains multiple conflicting labels then we ignore that voxel. We also examine two processes for determining the features of a voxel. As in [7] we can choose one of the points in the voxel at random to provide the features, or we can take the average of all the points features. This second method increases the computational overhead significantly as *torchsparse* does not provide support for this in there fast voxelization process. The same scene as viewed above can be seen voxelized in the figure below.

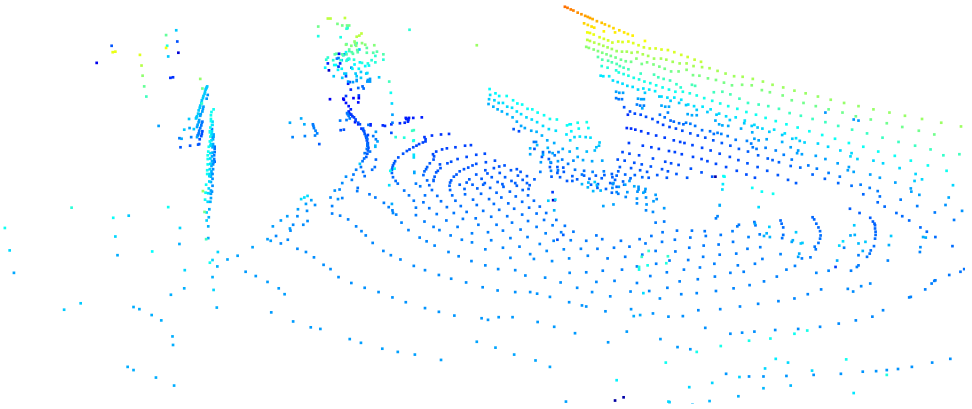

Figure 3: A heavily voxelized version of the scene.

The nuScenes dataset as 31 different classes of which we only take a subset. This is done somewhat due to the hierarchical structure, so we treat a child and an adult both as pedestrians, and also to simplify the problem.

4.2 Model

The SPVNAS model found by [7] is based on a U-Net style structure. Sparse point voxel convolutions are applied to the input to downsample, and is then upsampled with skip connections. We note that the point wise branches will wrap around multiple sparse convolutions. This model was pre-trained on the SemanticKITTI dataset.

We propose to use a recurrent module to combine the hidden state and the current input. However, a standard LSTM or GRU does not make senes for our uses as it applies dense operations. We propose using some form of convolution in the gates instead. The question is how? One might just apply sparse convolutions with or without an MLP wrapped around. Alternatively, one might apply sparse point voxel convolutions which is similar to having two branches one with a standard gating scheme and another with a convolutional gating scheme.

5 Results

Some stuff here

6 Discussion

Some stuff here

References

[1] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks, 2019.

[2] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks, 2017.

[3] Benjamin Graham and Laurens van der Maaten. Submanifold sparse convolutional networks, 2017.

[4] Rui Huang, Wanyue Zhang, Abhijit Kundu, Caroline Pantofaru, David A Ross, Thomas Funkhouser, and Alireza Fathi. An lstm approach to temporal 3d object detection in lidar point clouds, 2020.

[5] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. In *Advances in Neural Information Processing Systems*, 2019.

[6] Mahyar Najibi, Guangda Lai, Abhijit Kundu, Zhichao Lu, Vivek Rathod, Thomas Funkhouser, Caroline Pantofaru, David Ross, Larry S. Davis, and Alireza Fathi. Dops: Learning to detect 3d objects and predict their 3d shapes, 2020.

[7] Haotian* Tang, Zhijian* Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *European Conference on Computer Vision*, 2020.

[8] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection, 2017.

Appendix: