國立清華大學資訊工程系 111 學年度上學期專題報告

專題名稱	3D Instance Segmentation task				
參加競賽或計畫	☐ 參加對外競賽	☐ 參與其他計畫	[✓ 無參加對外競賽或任何計畫]		
學號	108062122	108048110			
姓名	顏訓哲	郭玫均			

摘要

Computer vision is one of the most popular subject in computer science.
While its applications are full of our daily life, we still have less opportunity to get inside of it.
In this project, we realize the basis of computer vision; furthermore, we also do some experiments in order to tune the model of 3d instance segmentation.

中華民國 111 年 11 月

Outline

Motivation and purpose

With the progress of both Artificial Intelligence and hardware architecture, many fields of computer science are able to develop significantly. Among these fields, the progress of computer vision is the most obvious to all.

Within these decades, Computer Vision has published a great amount of impressive applications, such as autonomous cars, face recognition, anomaly detection etc, which usually appear in our daily life.

Over the past years, remarkable advances in techniques for 3D point cloud understanding have greatly boosted performance. A lot of algorithms of analyzing point clouds, which sometimes have a satisfied result on simple tasks, have been published by many research teams and professors. Although these approaches achieve impressive results for object recognition and semantic segmentation, almost all of them are limited to extremely small 3D point clouds, and are difficult to be directly extended to large-scale point clouds.

Existing models usually perform not as expected when dealing with point clouds that have large amounts of instance, obviously different on scale of objects and numerous tiny objects, which limits the capability for computers to sense the 3D real world.

Computer vision is the most popular field nowadays, and we are still not familiar with related techniques such as machine learning and deep learning. So our project aims to realize the model of large scale 3D instance segmentation, and try different methods to improve its performance.

Model and Dataset

STPLS3D, Sematic Terrain Points Labeling – Synthetic 3D, is composed of both real-world and synthetic environments, which cover more than $17Km^2$ of the city landscape in the U.S. with the high quality of per-point annotations and with up to 18 fine-grained semantic classes and 14 instance classes, which include building, low vegetation, medium vegetation, high vegetation, vehicle, truck, aircraft, military vehicle, bike, motorcycle, light pole, street sign, clutter, fence.

figure 1

HAIS - Hierarchical Aggregation Instance Segmentation model intro

We can divide the overall architecture of the HAIS model into four parts, point-wise prediction, point aggregation, set aggregation and intra-instance prediction respectively.

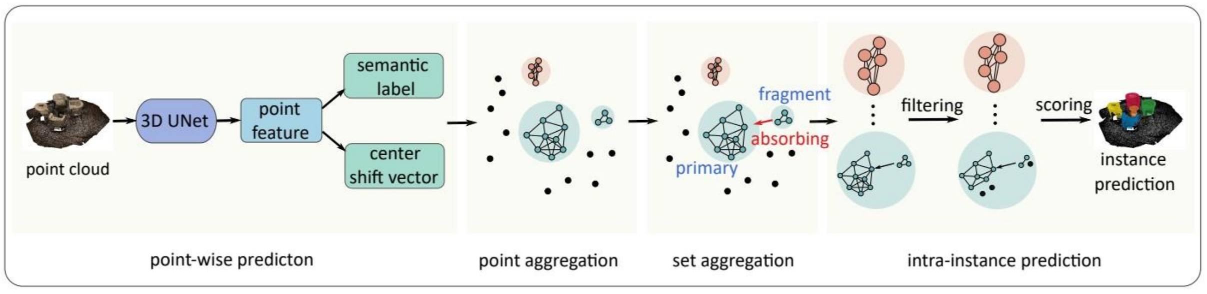


figure 2

Point-wise prediction network

The model first extracts features from the point clouds and utilize the coordinates and colors information to predict point-wise semantic labels and venter-shift vectors. Point-wise feature learning is performed with the submanifold spare convolution, which is widely used in 3-dimension perception methods to extract features from point clouds.

Submanifold sparse convolution

The conventional convolution feature map generally deals with dense data, but because of the sparsity in 3D point cloud, it is very inefficient for the traditional CNN to treat these spare matrices as dense data.

The difference between submanifold convolution and sparse convolution is that the latter will calculate an output site when its kernel covers an active input; while submanifold sparse convolution requires the venter of the kernel to cover an active input site before convolution output is calculated. In fact, the largest difference between manifold sparse convolution and sparse convolution is the calculation method of the rule book. There is not much difference in the prediction accuracy, but significance discrepancies between the memory overhead, cost and time.

After extracting features by performing submanifold convolution, the model converts the data into a volume mesh, and then uses a UNet-like architecture model, consisting of a stack of 3 dimension sparse convolution layers, to reel off voxels' attributes. By mapping the attributes back to point features, the model constructs two branches, one for predicting point labels and the other for predicting the center offset of each point.

__Semantic Label Prediction Branch__

This branch uses 2 layers of multilayer perception, which adopt gradient descent to obtain the best unbiased estimator, on the point features and a softmax layer to generate semantic scores for each class. The class with the highest score will be regarded as the predicted label of these points. Finally, the model utilizes cross-entropy to train this branch.

__Center Offset Prediction Branch__

This branch is parallel to the semantic label prediction branch, again it applies a 2-layer MLP on the point features to predict the point-wise center offset xi (xiR3), Lshift(loss) is used to optimize the prediction of center offset during training. Lastly, the instance center is defined as the average of all points in the instance.

$$
\begin{aligned}
&\mathcal{L}_{\text{shift}} = \frac{1}{\sum_{p_i \in P} \mathbb{1}(p_i \in P_{\text{fg}})} \cdot \sum_{p_i \in P} \mathcal{L}(p_i), \\
&\mathcal{L}(p_i) = w(p_i) \cdot \| \triangle x_i^{\text{gt}} - \triangle x_i^{\text{pred}} \|_1 \cdot \mathbb{1}(p_i \in P_{\text{fg}}), \\
&w(p_i) = \min(\| \triangle x_i^{\text{gt}} \|_2, 1).
\end{aligned}
$$

figure 3

(The closer the point is to the center of the instance, the smaller the value of the center shift vector, and therefore the less the loss contributed.)

Point Aggregation

Based on the fact that points with the same instance in the 3 dimensional space are essentially adjacent, the initial instance can be obtained through the simplest clustering method. First, according to the calculated point-wise center offsetxi, the model shifts xi to its center by xishift = xiorigin+xi, then it discards the background points and treats the foreground points as nodes. Afterward, for rpoint, which has the same semantic label and a fixed clustering bandwidth will divide a line between the two nodes, background and foreground. After traversing all nodes and establishing edges, the entire point cloud is divided into multiple independent sets. Each set is a preliminary instance prediction.

Set Aggregation

But point aggregation is not a guarantee to the fact that all points in an instance are grouped correctly. Most points with the precise center offset predictions can be clustered together to form an incomplete preliminary instance prediction, called primary instancel while a few points with poor center offset predictions are split from the majority. These fragments have few points that can not be considered as complete instances, but missing parts of the incomplete primary instances. Considering the massive amount of fragments, it is irrational to simply dropout these groups directly within the threshold. Meanwhile, intuitively, if we look at the groups at the set-level, we can aggregate the primary instance and the fragment group to form a complete instance prediction.

Intra-instance Prediction Network

Due to the fact that the previous aggregation process may erroneously absorb fragments belonging to other primary instances, resulting in flawed instance predictions, HAIS designs this network to further subdivide instances and filter out noisy data and outliers. First of all, the point clouds are cropped into 50*50*50 blocks and be fed as input, and then the 3D submanifold sparse convolution network is used to extract the instance attributes. After the intra-instance feature extraction, the mask branch will classify whether the point belongs to instance foreground or instance background (FG and BG are based on specific instance). For each predicted instance, the best matching ground truth is selected as mask supervision. In the end, the part where the predicted instance overlaps with the groundtruth is assigned to positive labels, while other parts are assigned to negative labels which only instances with IoU greater than 0.5 will be selected as training samples.

$$\mathcal{L}_{\text{mask}} = -\frac{1}{\sum_{i=1}^{N_{\text{ins}}} \mathbb{1}(iou_i > 0.5) \cdot N_i} \cdot \sum_{i=1}^{N_{\text{ins}}} \Big\{ \mathbb{1}(iou_i > 0.5) \cdot \sum_{j=1}^{N_i} [y_j \cdot \log(\hat{y}_j) + (1 - y_j) \cdot \log(1 - \hat{y}_j)] \Big\},$$

figure 4

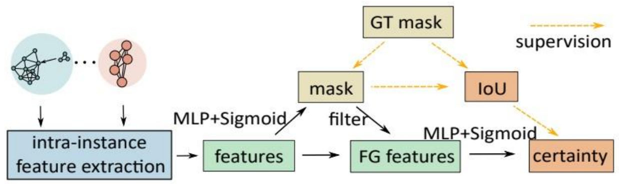
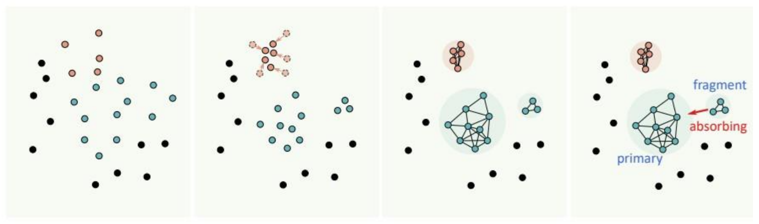


figure 5

(The mask is used to filter background points, as for the rest of the foreground points, they are sent to the sigmoid layer with MLP to predict the instance score. Secondly, we regard the IoU value calculated between the predicted mask and ground truth as the mask quality and use this quality to supervise the certainty of the instance.)



(a) original coordinates (b) shifted coordinates (c) point aggregation (d) set aggregation

figure 6

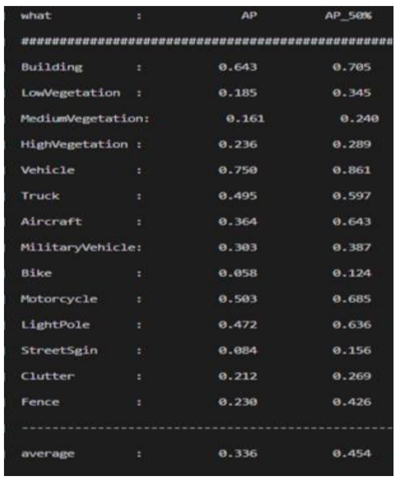
a. The distribution of points in the real 3D space, the points with different colors indicate that they belong to different categories.
b. After applying center shift vector branch at each point, the points which were predicted with the same instance label would essentially be closer in the dimensions.
c. Point aggregation, which aggregates points into sets based on a fixed spatial clustering bandwidth.
d. Set aggregation, primary instance will absorb the surrounding fragments with dynamic clustering bandwidth to form a complete instance.

Experiment

Introduction

Our environment runs on Ubuntu with version 18.04.6 LTS. Environment is composed of CPU, Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz, 64G Memory , and Nvidia-GTX-1080 8G.

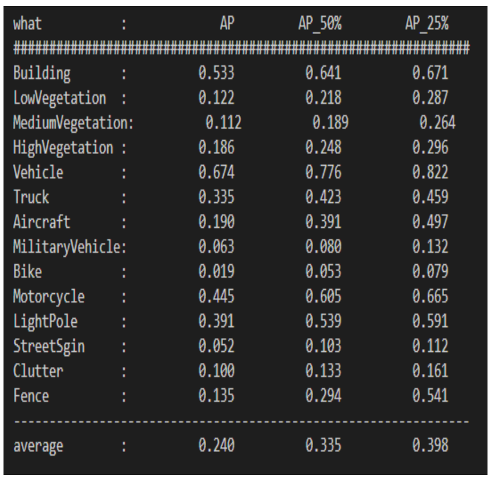
First, to start the experiment, we run the model without doing any modify as our baseline. The result record as following. According to the result, we can find that this model is bad at recognize vegetation and street sign.Original training time with 500 epochs is about 4 days. For the purpose of getting the result in a more efficient and fast way, we change the train epoch from 500 to 100, and the time for training reduce from about four days to one day.



```
what             :         AP        AP_50%
####################################################
Building         :      0.643         0.705
LowVegetation    :      0.185         0.345
MediumVegetation:        0.161         0.240
HighVegetation   :      0.236         0.289
Vehicle          :      0.750         0.861
Truck            :      0.495         0.597
Aircraft         :      0.364         0.643
MilitaryVehicle:        0.303         0.387
Bike             :      0.058         0.124
Motorcycle       :      0.503         0.685
LightPole        :      0.472         0.636
StreetSgin       :      0.084         0.156
Clutter          :      0.212         0.269
Fence            :      0.230         0.426
----------------------------------------------------
average          :      0.336         0.454
```

figure 7

(500 epoch)



```
what           :        AP        AP_50%       AP_25%
#######################################################
Building       :     0.533        0.641        0.671
LowVegetation  :     0.122        0.218        0.287
MediumVegetation:     0.112        0.189        0.264
HighVegetation :     0.186        0.248        0.296
Vehicle        :     0.674        0.776        0.822
Truck          :     0.335        0.423        0.459
Aircraft       :     0.190        0.391        0.497
MilitaryVehicle:     0.063        0.080        0.132
Bike           :     0.019        0.053        0.079
Motorcycle     :     0.445        0.605        0.665
LightPole      :     0.391        0.539        0.591
StreetSgin     :     0.052        0.103        0.112
Clutter        :     0.100        0.133        0.161
Fence          :     0.135        0.294        0.541
-------------------------------------------------------
average        :     0.240        0.335        0.398
```

figure 8

(200 epoch)

Data augmentation

The first method we try is data augmentation.

Sampling method:

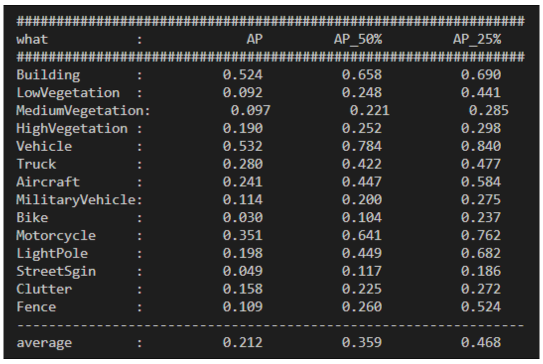
The data augmentation method used by the original HAIS model is to rotate point features and 3D coordinates through simple random sampling. However, we found that due to the limited number of samples, there is a high probability that the angle of rotation is actually only volatile within a small range.

Our solution to the problem is to use stratified random sampling and increase the number of samples. We divide the sampling unit by 60 degrees, divide the rotation angle into different levels, and then randomly select samples from these levels independently. This modification enlarges the range of rotation angles and improve the accuracy of the estimation.

Another way we try is to add noise. The noise we chose is salt and pepper, which we randomly add that noise into half of the dataset.

The result of our experiment is shown in figure 9. The ap is lower than baseline, but the ap 25 and ap 50 are higher than baseline. Further, we can observe that the prediction of instances with shape are higher than baseline, but the prediction are lower for tiny things or vegetation instance.

We assume that the main reason of the result is that the coverage of vegetation is high, and randomly noise have high probability distribute on vegetation, and feature tiny instances are hard to learn, which leads to the worse performance in model.



```
################################################################
what            :          AP        AP_50%        AP_25%
################################################################
Building        :       0.524        0.658         0.690
LowVegetation   :       0.092        0.248         0.441
MediumVegetation:        0.097        0.221         0.285
HighVegetation  :       0.190        0.252         0.298
Vehicle         :       0.532        0.784         0.840
Truck           :       0.280        0.422         0.477
Aircraft        :       0.241        0.447         0.584
MilitaryVehicle:        0.114        0.200         0.275
Bike            :       0.030        0.104         0.237
Motorcycle      :       0.351        0.641         0.762
LightPole       :       0.198        0.449         0.682
StreetSgin      :       0.049        0.117         0.186
Clutter         :       0.158        0.225         0.272
Fence           :       0.109        0.260         0.524
----------------------------------------------------------------
average         :       0.212        0.359         0.468
```

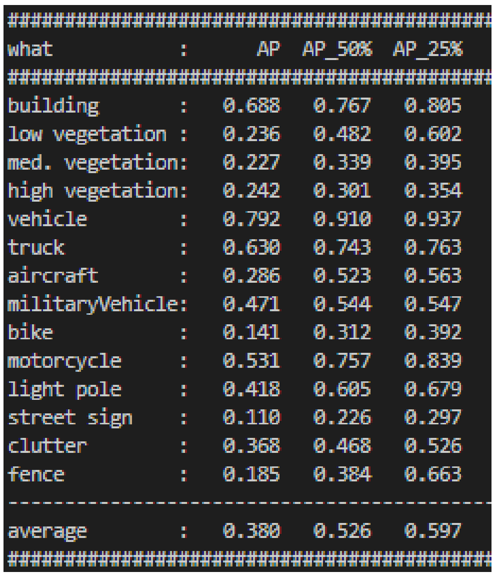
figure 9

Attention layer

Attention mechanism is like the way that how human observe the world. Human’s vision always put their limited attention on main point, which can not only save the resource but also fetch the effective data fast. This mechanism mainly solves the problem of data loss resulting from the data compressed form residual block and encoder-decoder. Attention aims to weight the input of the block, which can extract some critical data and let the model predict more precisely.

Since our model contains residual block, we decide to put attention layer into our model. The attention we choose is Squeeze-and-Excitation Networks, which has less computation, complexity and parameter but is obviously effective. Squeeze-and-Excitation Networks first apply a average pooling on input feature map, and get a one dimension vector. Second it apply two linear layer and two non-linear layer on the vector to get the weighted score of each channel. Last is cast the weighted score on corresponding channel.

The experiment result is shown in figure 10. As you can see, every prediction of an instance becomes much higher than the baseline. SE layer does help our model to focus on important information and extract more critical attributes so that the model can make more accurate judgments. The effect of the SE attention block is extraordinary. Despite the rising performance, the training time becomes longer than baseline by half of a day.



```
##############################################
what           :      AP  AP_50%  AP_25%
##############################################
building       :   0.688   0.767   0.805
low vegetation :   0.236   0.482   0.602
med. vegetation:   0.227   0.339   0.395
high vegetation:   0.242   0.301   0.354
vehicle        :   0.792   0.910   0.937
truck          :   0.630   0.743   0.763
aircraft       :   0.286   0.523   0.563
militaryVehicle:   0.471   0.544   0.547
bike           :   0.141   0.312   0.392
motorcycle     :   0.531   0.757   0.839
light pole     :   0.418   0.605   0.679
street sign    :   0.110   0.226   0.297
clutter        :   0.368   0.468   0.526
fence          :   0.185   0.384   0.663
----------------------------------------------
average        :   0.380   0.526   0.597
##############################################
```

figure 10

Parameter tune

The method we decide to prevent overfitting is by adding dropout layers to our model. According to the explanation provided by the official document, adding the dropout layer has the effect of taking the average to all tuned models. If we use the same training data to train 5 different neural networks, we will generally get 5 different results. At this point, we can use the "average of 5 results" or "majority-winning voting strategy" to decide the final result. For example, if 3 networks judge that the result is the number 9, then it is very likely that the real result is the number 9, and the other two networks give the wrong result. Because different networks may overfit differently, averaging them may offset some of the overfits, this "combination and averaging" strategy is often effective in preventing overfitting problems.

So, adding dropout is similar to training different networks. Randomly deleting half of the hidden neurons results in a different network structure. The entire dropout process is equivalent to averaging many different neural networks. Different networks have different overfitting problems while the advantage of adding additional dropout layers helps improve the prediction.

Next, adding dropout helps reduce the complex co-adaptation relationship between neurons. Since the dropout procedure changes the number of nodes appearing in every hidden layer, the update of weights no longer depends on the joint action of implicit nodes with a co-adaption relationship, which prevents some features from having perceptual effects only under certain node combinations. We force the network to learn more robust features that are also present in random subsets of other neurons. In other words, if our neural network is making some kind of prediction, it should not be too sensitive to some specific fragments, and even if specific data is lost, it should be able to learn some common features from many other cues. From this point of view, dropout is a bit like L1 and L2 regularization. Reducing the weight makes the network more robust to the loss of specific neuron connections.

After adding a dropout layer to our model's residual block, we obtain the result presented in table 1. Comparing the segmentation prediction accuracy to the baseline, the average precision decreases significantly instead, which contradicts our assumption. Since the number of parameters in the model is approximately 7.7 million, it is reasonable to dropout out some nodes. The only reason we get is the insufficient training epoch since dropping out nodes under such circumstances may lead to the model's poor understanding of the dataset.

Table 1

	AP	AP50
Dropout (0.5)	0.099	0.207
Dropout (0.7)	0.212	0.359
Dropout (0.8)	0.233	0.378

Conclusion

We joined the lab last summer and have been exploring and conducting experiments in the field since. We learned the basics of how a cluster-based model performs the instance segmentation task. Upon doing the experiment, we encountered a lot of technical problems, and among these problems, it is creating a virtual environment for the HAIS model that took the most time. To sum up, according to the previous experiment we conducted, adding new blocks into the model is more effective in the performance of the model than tuning parameters, but the cost is that it must spend more time on computing additional layer.

References

1. Hierarchical Aggregation for 3D Instance Segmentation
2. STPLS3D: A Large-Scale Synthetic and Real Aerial Photogrammetry 3D Point Cloud Dataset
3. Squeeze and Excitation attention
4. Softgroup
5. 3D U-Net
6. PyTorch Dropout