Image Detector Based Automatic 3D Data Labeling and Training for Vehicle Detection on Point Cloud

Zhengyong Chen[1], Qinghai Liao[2], Zhe Wang[1], Yang Liu[2] and Ming Liu[2]

***Abstract*— Nowadays, a large amount of labeled data is crucial for deep neural network training. However, data labeling is still a time- and labor-consuming task, especially when labeling 3D point clouds. Meanwhile, object recognition has achieved great success on 2D images, even beyond the ability of humans. In this paper, we propose an effective framework to produce labeled data by using an image detector as a supervisor, and we train the network with a simple trick to eliminate noisy labels. For object-sparse scenes, this method is able to obtain good label data, while for object-dense scenes, we can use our training method to detect some of the corrupted labels. This is realized by building a cohesive camera and LiDAR system (named "Licam") and performing target frustum region proposal on point clouds using the camera detection result. Efficient and effective vehicle detection is achieved based on this learning and training framework. We examine this method on the KITTI dataset [7] and our own road running data collected from a micro electro mechanical system (MEMS) LiDAR, demonstrating fast and accurate detection results. The results show that our automatic data labeling and training framework is effective and efficient. It provides the ability to obtain large-scale labeled data, and is easy to use for online learning.**

I. INTRODUCTION

Recently, 2D image recognition has had great achievements in areas such as image classification[1], object detection[2],[3] and semantic segmentation[4]. In certain image recognition tasks, machines can even work better than humans. Behind these achivement is data support from large-scale labeled datasets, like ImageNet[5], which has more than 100,000 sysnets and provides on average 1000 images to illustrate each synset. These sufficient data supply the strength to train deeper and deeper networks. However, for the increasingly popular 3D sensors, 3D object detection has not yet achieved a similar satisfactory result. One reason is the lack of enough labeled data to train a common 3D feature extraction model, like an ImageNet pre-tained VGG16[6] network. Taking the KITTI dataset[7] as an example, its 3D object detection data consists of only 7481 training point clouds, and the detection results of existing methods are still not as good as our expectation. For this problem, we propose a simple but effective method to automatically label data and do 3D object detection on KITTI dataset and the dataset we collected with our own MEMS LiDAR.

Data labeling is a time- and labor- consuming task, especially 3D data labeling. Within 3D data labeling, point cloud data labeling has been extensively studied under many different situations. Most of the approaches label point clouds by training a graphical model, or a classifier captures various features from contextual, spectral, prior knowledge and geometrical information(e.g., [10],[11]). Some researchers, like the authors of [12], have also proposed a detection-based approach for labeling in reconstructed 3D point clouds. But these methods are based on hand-crafted features and cannot provide satisfactory accuracy and speed. In this work, we set up a cohesive system with a camera and MEMS LiDAR, which we call "Licam", so we can easily and accurately reproject the image detection result into the point cloud and finally obtain the target frustum region (as shown in Fig. 3). We choose the SSD-detector[3] for image detection. Since it is more accurate and can be run real-time, which is beneficial for online learning. In addition, our MEMS LiDAR is cheap, dense and able to produce dense and abundant point cloud data. This makes producing massive labeled data possible.

But the labeling results produced by reprojecting image detection result to point cloud are not able to fit the needs of deep learning training data. Because when there is label noise, they tend to over-fit[18]. For any classification, the cornerstone of deep learning based detection models, a degradation in performance is inevitable when there is noise in the training data. To overcome this problem, researchers have mainly focused on loss function tolerance analysis[18],[25] and learning noise distributions to compensate the network[24],[26],[27]. For this problem, we design a training strategy to use a pre-trained network to detect and avoid corrupted data, and it works for binary classification.

Researchers are still exploring how to use deep learning to do object detection with 3D point cloud data. Most existing deep learning methods project 3D point clouds as 2D images and do image detection by fine-tuning image pre-trained models(e.g., [13]). Some researchers have also used a supervised 3D convolutional neural network(CNN) to process point clouds (e.g., [15], [16]). Different from previous methods, Qi et al. proposed a 3D object detection model which processes raw point clouds and images directly. But this approach is heavily dependent on image detection results and sometimes does not work, such as in night-time situations. In this paper, we propose an approach which only

*This work was supported by National Natural Science Foundation of China No. U1713211 and 61603376, the Shenzhen Science, Technology and Innovation Commission(SZSTI) JCYJ20170818153518789, JCYJ20160428154842603, the Research Grant Council of Hong Kong SAR Government, China, under Project No. 11210017 and No. 21202816, and was also supported by the Guangdong Innovation and Technology Fund No. 2018B050502009, awarded to Prof. Ming Liu and Dr. Lujia Wang.

[1]Zhengyong Chen, Zhe Wang are with Unity-Drive Innovation (UDI) Ltd. {chenzhengyong,wangzhe}@unity-drive.com

[2]Qinghai Liao, Yang Liu and Ming Liu are with Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong SAR, China. {qhliao,yang.liu,eelium}@ust.hk

uses an image detector as a supervisor of a PointNet++[8] classifier and works for target frustum region proposal and data labeling. We propose an "auto-min-cut" segmentation method and feed clusters into the classifier. Moreover, due to the density of MEMS point cloud data, the model can more easily do detection.

The main contributions of this paper are as follows:

- We propose an efficient auto-labeling and segmentation framework that automatically and precisely labels 3D point cloud data.
- We propose a method to detect and avoid corrupted data for binary classification.
- To the best our knowledge, this is the first use of auto-labeled point cloud data to train a deep neural network. We implement image detector-labeled vehicle and pedestrian detection on our MEMS LiDAR.

II. Related Work

A. Automatic Labeling

3D data labeling has been extensively studied over the years. Previous works have focused on: using hand-crafted geometrical and spectral features from a single LiDAR sensor[11] and integrating image information and point cloud features[10],[12]. Anand et al.[10] used a graphical model that captures various features and contextual relations from visual appearance and some prior knowledge. Lai et al. [12] meanwhile, re-project image detection results into the point cloud, labeling objects in a reconstructed 3D scene and using an markov random filed (MRF) model to do segmentation. The point cloud data is collected by a kinetic-style sensor, and the object is rotated on a turntable, but this costs large amounts of money and time. The above approaches do not provide sufficient results to use as training data for deep learning.

B. Training with Noisy Labels

In every classification model, degradation of performance is inevitable when there is noise in the training labels[24]. The simplest approach is to remove the corrupted data manually. However, this is unacceptable when dealing with large-scale data. To address this problem, researchers have mainly focused on loss function promotion and noise distribution learning. [18] analyzed the noise tolerance properties of different loss functions under risk minimization. They found that the 0-1 loss function has impressive noise tolerance. [25] proposed two approaches to optimize the loss function to improve the noise tolerance of the model, and the randomly labeled noise is class-conditional in the paper. Finally, [24] and [26] introduced an extra noisy layer and an asymmetric bernoulli noise model into the network respectively to match the noisy label distribution, but their results are not very good.

C. Object Detection on Point Cloud with Deep Learning

There are many outstanding works on object detection on point clouds with deep learning. These approaches can be simply classified into two directions. The first is transforming 3D data to 2D images from different views, and using relatively mature 2D image based models, such as MV3D[13], to do detection. In the same direction, some approaches transform mature 2D image based techniques to 3D, like Vote3Deep[15] and VoxNet[16], using 3D convolution to extract 3D features of the point cloud. The second direction is to develop a specific model or technique to deal with point clouds, like Frustum PointNet[17], which utilizes the previously proposed method PointNet[14] to extract 3D features, together with image information, achieving the state-of-the-art. In our work, the automatic labeling and noise-robust training framework can help improve these models.

III. Approach

Our target is to automatically label data and train a point cloud classifier to implement vehicle detection. The first task is to set up the Licam, which serves to automatically label data. The next task is to train the network with these noisy data. The input to our final model is a point cloud cluster, which can be produced by segmentation methods such as Euclidean segmentation[19] and min-cut segmentation[9]. Finally, we apply this point cloud classifier to vehicle detection combined with Euclidean segmentation. The whole process is illustrated in Fig. 2. We describe each component below.

A. Automatic Point Cloud Labeling

We re-project the detection result from the image detector onto the point cloud, and from here the frustum region of interest is proposed. This region with labels is then refined by further segmentation.

Image Detector: Currently, there are plenty of image detection models, such as Faster-RCNN[20], R-FCN[21], SSD-network[3], YOLO[22] etc. The detection results can be trusted because these deep neural networks usually have a low false detection rate. Here we choose SSD-network as the image detector. We only use the detected point clouds as positive samples, so mis-detection will not have an effect on our method.

Cohesive Camera & LiDAR System: The basis of our automatic point cloud labeling is the Licam. With this system, we can get RGB-D data easily, incorporating the results of an image based model and point cloud based model. The constructed MEMS LiDAR-camera system is shown in Fig. 3(illustrating the MEMS and camera, the related position and the fixed device).

Assuming we already have the image detection result, through the flow diagram provided in Fig. 2 we can crop the corresponding point clouds easily and coordinate system transformations as shown in Fig. 1. One can easily conclude that the more close the centers of the camera and LiDAR are, the more precise the re-projection result will be.

B. Frustum Proposal and Refined Segmentation.

Target Object Point Cloud Proposal: The region of interest is proposed after the object is detected by an image detector. Now the frustum region is isolated but still contains some

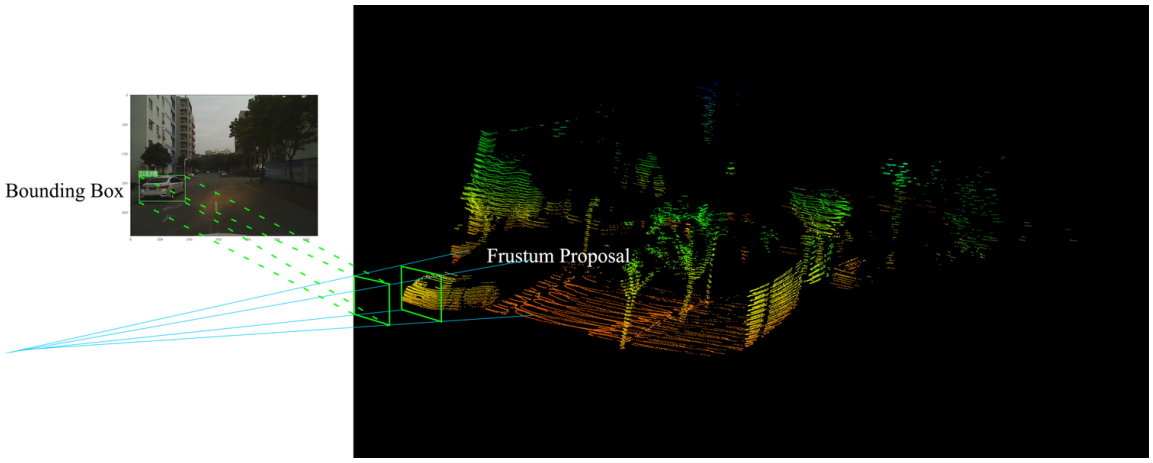


Fig. 1: Graph illustrating the process of transformation from the image detector's detection box to frustum proposal in the point cloud.

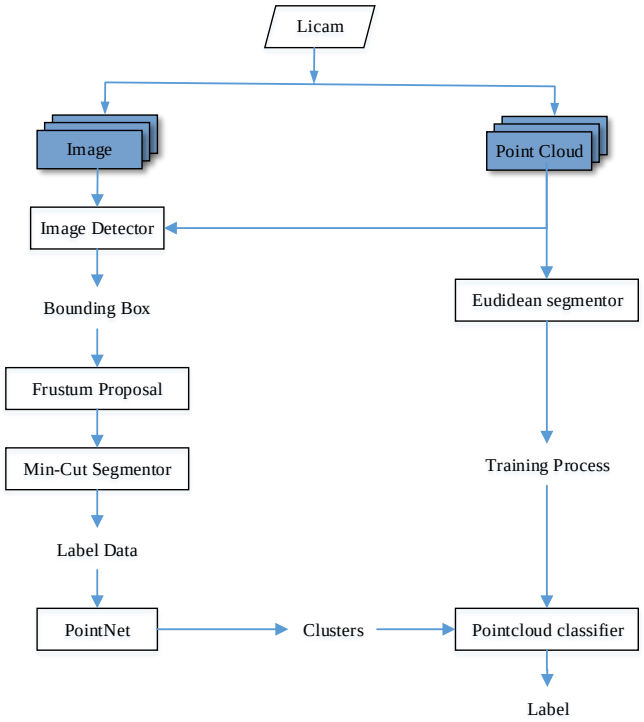


Fig. 2: Process of automatically labeling and training network with noisy labels for vehicle detection.

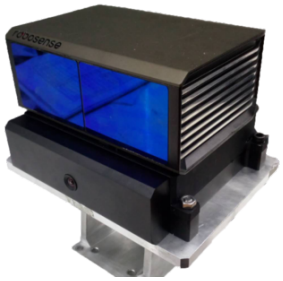


Fig. 3: Cohesive MEMS LiDAR and camera system, which we call a "Licam". The blue area is the MEMS LiDAR, and the bottom is the camera.

background points, as shown as Fig. 4c and Fig. 4g. In this process, a seed point which is used for min-cut segmentation is found after reprojecting the detection result into the point clouds:

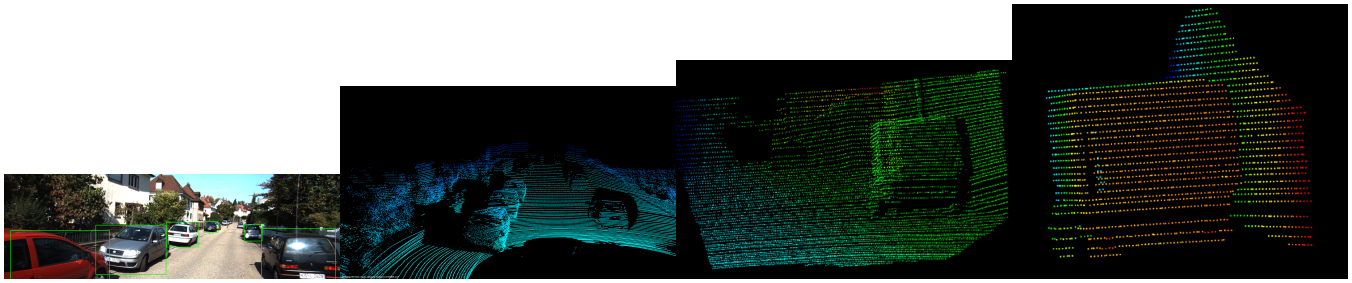
The min-cut segmentation algorithm[9] builds a k-nearest neighbors graph, assumes a background priority, adds hard foreground constraints, and finds the min-cut to compute a foreground-background segmentation. These features of the frustum region(foreground is isolated and occupies a large part of the point cloud) are well suited to using min-cut segmentation to extract the refined foreground. The result of min-cut segmentation is very clean if all the factors are satisfactory, as shown in Fig. 4d and Fig. 4h.

Background Point Cloud Proposal: In our method, we only find the image which does not contain the target object. Or we can use a self-designed UI to point out those regions which are not nearby the target object. These points are used as seed points projection on the image. The corresponding point clouds of the background images and the seed points can also follow the above steps to produce background clusters.

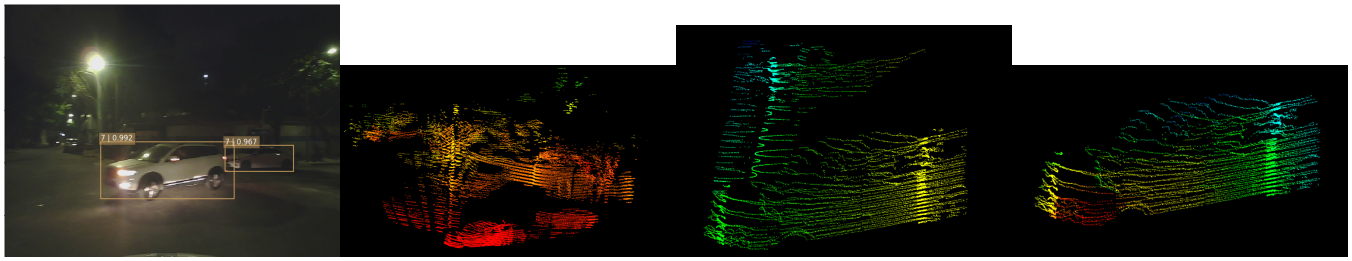
C. Vehicle Detection

In this paper, we focus on vehicle detection, so which is a binary classfication problem. First of all, we use the euclidean segmentation[19] method to segment the point cloud to clusters. Then a PointNet++[8] classification network is used to infer a tag to the cluster. The process is shown in Fig. 2.

The original PointNet++ was designed for point clouds with a fixed number of points, but the numbers of points of our clusters are different. The network only processes the clusters with sizes less than N, and employs zero-padding for these point clouds.



(a) KITTI, Detection (b) KITTI, Point Cloud (c) KITTI, Frustum Proposal (d) KITTI, Min-Cut Segmentation



(e) MEMS, Detection (f) MEMS, Point Cloud (g) MEMS, Frustrum Proposal (h) MEMS, Min-Cut Segmentation

Fig. 4: Different states' results of frustum proposal on the KITTI dataset and MEMS point cloud.

D. Noisy Labels

Because of the complexity of the environment and the defects of models, several types of noise, as listed in the following, will appear when using our proposed method, which will lead to corrupted data being extracted.

Mislabeling: As we mentioned before, the labels of the data come from the image detector. Therefore, some label noise will be caused by the image detector, as shown in Fig. 5a. But the classification accuracy of image detection is relatively high, so this problem will occur rarely.

Shade from Other Objects: When the scene is object-sparse, like a car running on a high-speed road, the foreground is isolated and there is a lack of other objects in front of the target object. The segmentation results will be like those in Fig. 4d and Fig. 4h. Conversely, if the scene is object-dense, the problem of shade from other objects will occur. This will cause a serious problem, like that shown in Fig. 5b, where the background object is labeled with the foreground tag.

Connective Object: This noise is caused by the segmentation algorithm. As mentioned before, min-cut segmentation uses some hard foreground constraints, like the max radius of the foreground range, and sometimes the parameter is not very precise or the object's body is incomplete, as shown in Fig. 5c. In this situation, the refined segmentation result will cause some background objects to be included.

Both mislabeled noise and connective objective noise appear rarely in object-sparse scenes, while noise of shade from other objects will occur frequently when the scene is object-dense.

E. Training with Noisy Label

For noisy labels, we focus on conquering the corrupted data caused by shade from other objects. Since our network is a binary classifier, the background samples are generated from a non-object point cloud. Therefore, the corrupted data will appear in positive samples only, which is the first feature of our noisy data. The second feature is that the corrupted data are those background samples with vehicle labels, and the third is that there is a small amount of corrupted data in our training data.

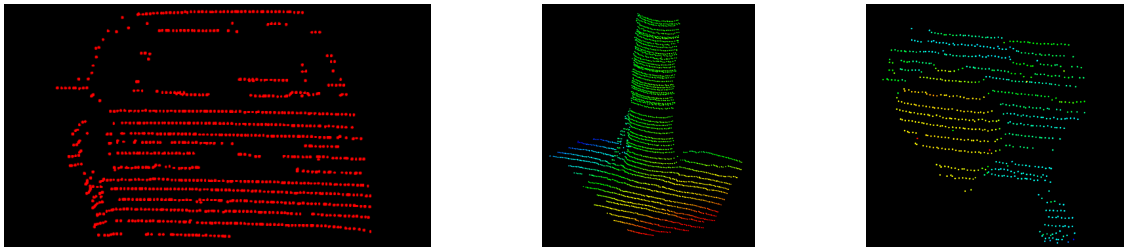
According to these features, we propose an efficient method to train a deep network and detect wrong positive samples. As shown in Fig. 6, PointNet will use the whole training data to train the network over several epochs. In this pre-training process, we assume the network has the basic ability to identify the easy samples. Then the network will find those positive samples whose classfication score of belonging to the negative samples is bigger than a certain threshold, and these sample will be removed from the training queue. This method works effectively in the actual training process.

IV. Experiment

To evaluate our method we consider data from two different datasets: our self-collected MEMS LiDAR dataset and the KITTI dataset(collected by Velodyne-64E LiDAR). Fig. 4b and Fig. 4f show examples of each.

A. Sensors and Data

MEMS LiDAR: The MEMS LiDAR we use is an HD solid-state LiDAR. This new MEMS-type LiDAR collects in real-time 3D high-precision distance information to help vehicles better perceive their surroundings with 100 channels.



(a) Mislabeling with "person" tag (b) Shade from other objects (c) Connective object

Fig. 5: Three types of noise we met.

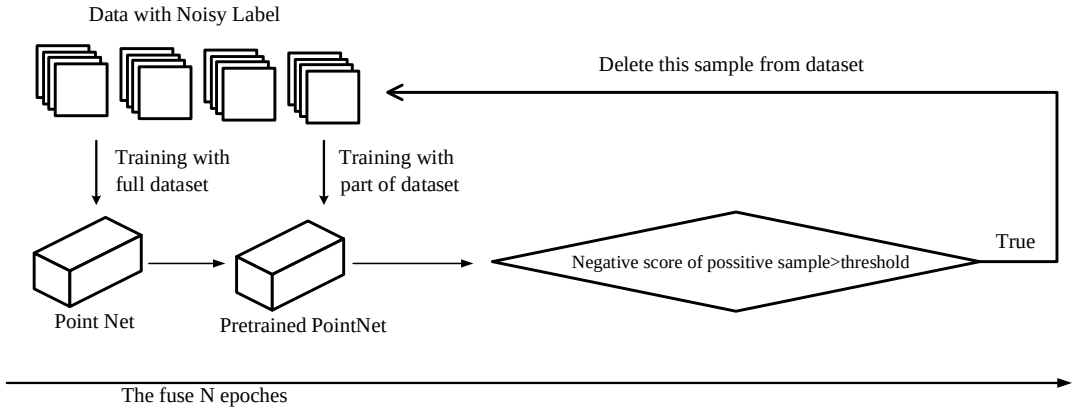
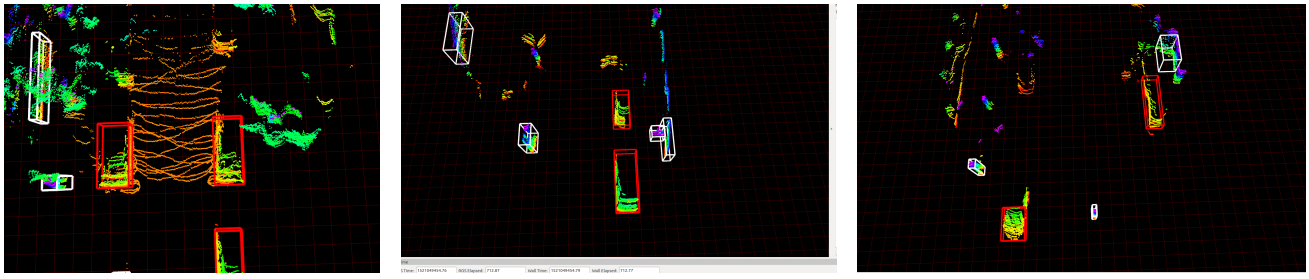


Fig. 6: PointNet training with noisy label.



(a) Kitii, detection (b) Kitii, point cloud (c) KITTI, frustum proposal

Fig. 7: Vehicle detection results on MEMS point cloud.

We set up our Licam on the top of the test car, with the height of the test car from the ground is 1.95m. We collect data ranging from suburban to city roads.

KITTI Dataset: The 3D object detection benchmark KITTI dataset consists of 7481 training images and 7518 test images as well as the corresponding point clouds, comprising a total of 80.256 labeled objects in an image on average. For validating our automatic labeling method, we do not use the labels, but utilize the provided training dataset and split it into training and testing sets for evaluation.

B. Frustum Region Proposal

Image Detector: With the Licam, we collect images with resolution 1280x720. We follow the timestamps to find the best matching image and point cloud. For this resolution, we choose an SSD-network as our image detector since the accuracy of the bounding box(shown in Fig. 4e) and the inference speed of the SSD-network actually meets our needs for precise labeling and online learning respectively.

In the KITTI dataset, the size of the collected and cropped images is 1224x370. We choose KITTIBox as the image detector for the KITTI dataset. KITTIBox is a collection of scripts to train FastBox, proposed by MultiNet[23]. KITTI-Box has higher accuracy than Faster-RCNN with a high fps of up to 27.97.

Frustum Proposal: After calibrating our Licam, we crop the frustum region according to the image detection result. For the KITTI dataset, we use the provided calibration results. We simply set a "z" axis filter for ground removal, and employ min-cut segmentation to refine the detected point

cloud. For negative samples, the background, we simply pick the point clouds corresponding to the non-vehicle images, and adopt Euclidean segmentation to generate background clusters.

The point clouds produced in this process are shown in Fig. 4c and Fig. 4g. The results of frustum region proposal are clean and complete, but some noisy labels arise with crowded scenes in the KITTI dataset.

C. Training with Noisy Labels

The data with noisy labels are used to train a PointNet++ classifier. We adopt zero-padding to set the input as a fixed number of point clouds. The labels and the prediction are represented as one-hot code. The loss function in this model is sigmoid cross entropy.

The vehicle detection results are shown in Fig. 7a, Fig. 7b and Fig. 7c. We can conclude that the measures we adopt to detect and avoid corrupted data do work.

The method proposed in this paper has not been tested in extreme environments, such as heavy rain, heavy snow and heavy fog. The noise generated by such weather is determined by the performance of the sensors and the image detectors, so we have not done the relevant experiments.

D. Vehicle Detection

Our vehicle detection process is very simple: use the configuration of background segmentation to deal with each coming point cloud, take each cluster in the point cloud as input, and the PointNet++ classifier outputs the corresponding label.

The detection results are shown in Fig. 7a, Fig. 7b and Fig. 7c.

E. Timing

We use a GEFORCE GTX1080-Ti in our experiment for the PointNet++ classifier only. The CPU is an Intel i7-7700K. Each frame of the Licam dataset has 75,000 points, and there are 130,000 points per frame in the KITTI dataset. The time consumption of segmentation and classfication are 93.68 ms and 83.33 ms respectively.

V. Conclusions

In this work, we have presented a framework which automatically and efficiently labels and trains with noisy labels, based on our Licam. We tested this framework on vehicle detection, and it shows good properties such as precise and automatic labeling and robustness to noise.

Obviously, this framework has the potential to learn online with large-scale labeled data. In the future, we are interested in applying this framework to learn the normal feature representation of point clouds.

References

[1] He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition[J]. computer vision and pattern recognition, 2016: 770-778.

[2] Ren S, He K, Girshick R B, et al. Faster R-CNN: towards real-time object detection with region proposal networks[C]. neural information processing systems, 2015: 91-99.

[3] Liu W, Anguelov D, Erhan D, et al. SSD: Single Shot MultiBox Detector[J]. european conference on computer vision, 2016: 21-37.

[4] He K, Gkioxari G, Dollar P, et al. Mask R-CNN[J]. international conference on computer vision, 2017: 2980-2988.

[5] Deng J, Dong W, Socher R, et al. ImageNet: A large-scale hierarchical image database[C]. computer vision and pattern recognition, 2009: 248-255.

[6] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition[J]. international conference on learning representations, 2015.

[7] Geiger A, Lenz P, Stiller C, et al. Vision meets robotics: The KITTI dataset[J]. The International Journal of Robotics Research, 2013, 32(11): 1231-1237.

[8] Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. (2017)

[9] Golovinskiy A, Funkhouser T A. Min-cut based segmentation of point clouds[C]. international conference on computer vision, 2009: 39-46.

[10] Anand A, Koppula H S, Joachims T, et al. Contextually Guided Semantic Labeling and Search for 3D Point Clouds[J]. arXiv: Robotics, 2011.

[11] Ramiya A M, Nidamanuri R R, Krishnan R, et al. Semantic labelling of urban point cloud data[J]. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2014: 907-911.

[12] Lai K, Bo L, Ren X, et al. Detection-based object labeling in 3D scenes[C]. international conference on robotics and automation, 2012: 1330-1337.

[13] Chen X, Ma H, Wan J, et al. Multi-view 3D Object Detection Network for Autonomous Driving[J]. computer vision and pattern recognition, 2017: 6526-6534.

[14] Charles R Q, Su H, Kaichun M, et al. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation[J]. computer vision and pattern recognition, 2017: 77-85.

[15] Engelcke M, Rao D, Wang D Z, et al. Vote3Deep: Fast object detection in 3D point clouds using efficient convolutional neural networks[J]. international conference on robotics and automation, 2017: 1355-1361.

[16] Maturana D, Scherer S. VoxNet: A 3D Convolutional Neural Network for real-time object recognition[C]. intelligent robots and systems, 2015: 922-928.

[17] Qi C R, Liu W, Wu C, et al. Frustum PointNets for 3D Object Detection From RGB-D Data[J]. computer vision and pattern recognition, 2018: 918-927.

[18] Manwani N, Sastry P S. Noise Tolerance Under Risk Minimization[J]. IEEE Transactions on Systems, Man, and Cybernetics, 2013, 43(3): 1146-1151.

[19] Rusu R B, Cousins S. 3D is here: Point Cloud Library (PCL)[C]. international conference on robotics and automation, 2011: 1-4.

[20] Ren S, He K, Girshick R B, et al. Faster R-CNN: towards real-time object detection with region proposal networks[C]. neural information processing systems, 2015: 91-99.

[21] Dai J, Li Y, He K, et al. R-FCN: Object Detection via Region-based Fully Convolutional Networks[J]. neural information processing systems, 2016: 379-387.

[22] Redmon J, Divvala S K, Girshick R B, et al. You Only Look Once: Unified, Real-Time Object Detection[J]. computer vision and pattern recognition, 2016: 779-788.

[23] Teichmann M, Weber M, Zollner J M, et al. MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving[J]. ieee intelligent vehicles symposium, 2018: 1013-1020.

[24] Sukhbaatar, S., Fergus, R.: Learning from noisy labels with deep neural networks. Eprint Arxiv (2014)

[25] Natarajan N, Dhillon I S, Ravikumar P, et al. Learning with Noisy Labels[C]. neural information processing systems, 2013: 1196-1204.

[26] Sukhbaatar S, Bruna J, Paluri M, et al. Training Convolutional Networks with Noisy Labels[J]. arXiv: Computer Vision and Pattern Recognition, 2014.

[27] Mnih V, Hinton G E. Learning to Label Aerial Images from Noisy Data[C]. international conference on machine learning, 2012: 203-210.