DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2019

Deep Active Learning for 3D Object Detection for Autonomous Driving

XIAO WEI

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Deep Active Learning for 3D Object Detection for Autonomous Driving

XIAO WEI

Master in Systems, Control and Robotics
Date: November 22, 2019
Supervisor: Atsuto Maki
Examiner: John Folkesson
Swedish title: Djup aktiv inlärning för 3D-objektdetektion vid autonom körning
School of Electrical Engineering and Computer Science

Abstract

3D object detection is vital for autonomous driving. However, to train a 3D detector often requires a huge amount of labeled data which are extremely expensive and tedious to obtain. In order to alleviate the annotation effort while maintaining detection performance, we aim to adopt active learning framework for training a 3D object detector with the least amount of labeled data. In contrast with the conventional passive learning that a machine learning model is trained on a pre-determined training dataset, active learning allows the model to actively select the most informative samples for labeling and add them to the training set. Therefore, only a fraction of data need to be labeled. To the best of our knowledge, this thesis is the first that studies active learning for 3D detection.

We take progressive steps towards the goal. There are three stages with increasingly complex models and learning tasks. First, we start with active learning for image classification which can be viewed as a sub-problem of object detection. Second, we investigate and build a multi-task active learning framework with a deep refinement network for multi-modal 3D object detection. Lastly, we further analyze multi-task active learning with a more complicated two-stage 3D LiDAR vehicle detector. In our experiments, we study the fundamental and important aspects of an active learning framework with an emphasis on evaluating several popular data selection strategies based on prediction uncertainty. Without bells and whistles, we successfully propose an active learning framework for 3D object detection using 3D LiDAR point clouds and accurate 2D image proposals that saves up to 60% of labeled data on a public dataset. In the end, we also discuss some underlying challenges on this topic from both academic and industrial perspectives.

Sammanfattning

Detektering av 3D-objekt är avgörande för autonom körning. För att träna en 3D-detektor krävs emellertid ofta en enorm mängd märkta data som är extremt dyra och omständiga att erhålla. För att underlätta anteckningsarbetet samtidigt som detekteringsprestanda bibehålls, använder vi ett aktivt lärande ramverk för att träna en 3D-objektdetektor med minsta mängd märkta data. I motsats till den konventionella passiva inlärningen som en maskininlärningsmodell tränas på i ett förutbestämt träningsdataset, tillåter aktiv inlärning modellen att aktivt välja de mest informativa proverna för märkning och lägga till dem i träningsuppsättningen. Därför behöver bara en bråkdel av data märkas. Såvitt vi vet är detta examensarbete det första som studerar aktivt lärande för 3D-upptäckt.

Vi tar steg mot målet. Det finns tre steg med allt mer komplexa modeller och inlärningsuppgifter. Vi börjar med aktivt lärande för bildklassificering som kan ses som ett delproblem för objektdetektering. För det andra undersöker och bygger vi ett multiverksamhetslärande ramverk med ett djupt förfiningsnätverk för upptäckt av flermodala 3D-objekt. Slutligen analyserar vi ytterligare flerfunktionsaktivt lärande med en mer komplicerad tvåstegs 3D-LiDAR fordonsdetektor. I våra experiment studerar vi de grundläggande och viktiga aspekterna av ett aktivt inlärningsramverk med tonvikt på att utvärdera flera populära datavalsstrategier baserade på förutsägelsosäkerhet. Vi föreslår framgångsrikt ett aktivt inlärningsramverk för 3D-objektdetektering med 3D-LiDAR punktmoln och exakta 2D-bildförslag som sparar upp till 60% av märkta data i ett offentligt dataset. Till slut diskuterar vi också några underliggande utmaningar i detta ämne ur både akademiska och industriella perspektiv.

Acknowledgements

This master thesis was registered at the KTH School of Electrical Engineering and Computer Science and accomplished at the Corporate Research Center of Robert Bosch GmbH.

First of all, I would like to thank Dr. Lars Rosenbaum and Mr. Di Feng, who are my supervisor and adviser from Bosch, for offering me this wonderful opportunity to study and research with the Driver Assistance Systems and Automated Driving team at the beautiful campus of Renningen.

I would like to express my special thanks to Mr. Di Feng, who has guided me in many aspects that I have ever hoped for. It was his insightful suggestions that walked me through the hardest time of the thesis. He has lively set up a model of intelligence and diligence that will keep inspiring me. It was truly a great time working with Mr. Feng as a mentee and a friend.

Also, I am grateful to my KTH supervisor Prof. Dr. Atsuto Maki who enlightened me on how to define and explore a scientific problem and reviewed my thesis with great patience. Prof. Maki impressed me not only as a scholar, but also as a man of humility and courtesy. It was my pleasure to accomplish this thesis under his guidance.

This thesis was conducted overseas and outside academy, which made it more difficult to communicate and manage sometimes. I am thankful to Prof. Maki, Prof. Dr. John Folkesson, Dr. Rosenbaum, Prof. Dr. Dongheui Lee, Ms. Sara Sjögren, Ms. Cristina La Verde, and many others who helpled me on administrative issues to make this thesis come to a successful end.

Furthermore, I appreciate the "Lunch Roulette" at Bosch, which granted me chances to meet domain experts of different backgrounds and have inspiring discussions on the autonomous driving industry.

Last but not least, I would like to thank my parents. They are ordinary people. But without their extraordinary love, I can never be who I am and achieve what I achieve today. I am also deeply thankful to my girlfriend, Qianzi, for her invaluable love and belief in me. While a machine learning model still actively searching for the answers from data based on uncertainty, I already know for certain that the answer I have been looking for is her.

Contents

List of Figures

Chapter 1

Introduction

In recent decades, autonomous driving has made significant progress in both the academy and industry [8][5]. It is expected to remove human error from driving, smooth traffic flow, save time for transportation, provide mobility options to the disabled, etc[1]. Towards achieving fully automated driving in all circumstances with the least human interventions, great efforts have been made to develop an accurate, robust, and interpretable environment perception system that is able to detect traffic agents, road signs, and other objects. The perception system usually deploys various kinds of sensors, such as LiDAR and camera, to locate and recognize surrounding objects. In practice, sensor data are independently analyzed or fused for object detection. In either case, a gigantic amount of labeled data is required to train a well-performed object detector. Despite collecting massive raw data from field tests is relatively easy, the cost of 3D annotation is very high, which hinders the development and implementation of autonomous driving technology. Consequently, training a 3D object detector to maximal performance with minimum expenditure becomes an inevitable problem for the autonomous driving industry.

1.1 Problem Statement

To train an accurate and robust 3D vehicle detector requires a vast amount of labeled data. The reason lies in two-fold. On the one hand, the detector needs a large comprehensive training set to cover as many objects and environments as possible which vary dearly in terms of type, appearance, pose, size, illumination, occlusion, etc. On the other

hand, a state-of-the-art 3D object detector is usually composed of deep artificial neural networks[1] which are naturally thirsty for data. Labeling an enormous quantity of raw data in 3D space, especially 3D LiDAR point clouds, demands huge human efforts, which are expensive and tedious.

To reduce the labeling cost for training while achieving high accuracy, it is desirable to train a 3D detector with as few labeled instances as possible. Intuitively, among the collected raw data, some are more representative and more informative to the detector. For instance, pristine samples with a clearer representation of target objects contribute better to feature extraction; corner case samples are very helpful to improve model generalization; etc. In contrast, some data are less informative or redundant, such as a chronological sequence of samples capturing almost the same view. Therefore, the problem of training a 3D object detector in a data-efficient manner can be transformed into how to select more informative data for training. One promising solution is active learning that enables an object detector to actively learn from the data. It has been studied with various machine learning tasks and has the potential to be extended to 3D detection. Nevertheless, active learning assumes that labeled data are limited, while unlabeled data are massive and accessible. In the case of 3D detection for autonomous driving, such an assumption is perfectly satisfied, as even a single autonomous driving vehicle is able to collect around 11 - 152 TB raw data per day[24]. In the next sections, we introduce the background and review related works on active learning and 3D detection. More details on our methodology are illustrated in Chapter 2.

1.2 Background

1.2.1 Active Learning

Active learning is an iterative framework that aims to make machine learning more economical. It participates in the acquisition of its own training data by actively selecting the most informative samples from unlabeled data for an oracle to annotate. Fig.1.1 shows an ideal active learning curve. To reach the same level of performance, an active learner shall require less labeling effort. Or given the same amount of

[1] Hereby, we denote artificial neural network by neural network for simplicity.

labeled data, an active learner shall achieve better performance. Work over the last few decades has shown that active learning is effective at maintaining accuracy while reducing annotated data size in many machine learning applications, for instance, speech recognition [45], information retrieval [57], image classification [19], etc.

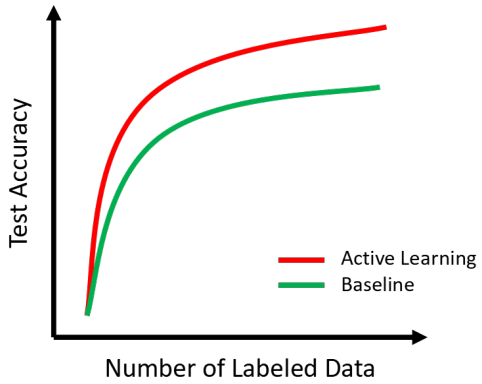


Figure 1.1: An ideal active learning curve. X-axis denotes the amount of labeled data used for training, y-axis denotes the test accuracy of the learner. Active learning aims to gain accuracy as fast as possible while minimizing the amount of annotation.

In general, there are three problem scenarios for active learning: membership query synthesis (unlabeled data typically generated by the system itself), stream-based selective sampling (unlabeled data come in on the fly), and pool-based sampling (unlabeled data have been collected in a pool) [49]. In this thesis, we focus on the last scenario and assume that all data needed have been gathered beforehand and available for evaluation, so that our work prioritizes on the study of the algorithm rather than data collection.

The iterative workflow of pool-based active learning is shown in Fig.1.2. We first initialize the active learner, who performs the desired machine learning task. Then the learner estimates the informativeness of each sample from the unlabeled data pool and selects the most informative ones, i.e. query data, for annotation. An oracle (usually a human) labels the query data and adds them to the training dataset. The learner is re-trained on the updated dataset, and a single iteration of active learning is finished. The next iteration starts with another round of informativeness estimation on the remaining unlabeled data and repeats the previous steps. The loop usually stops when the

learner reaches a certain level of performance or the unlabeled data pool is exhausted by iterative queries.

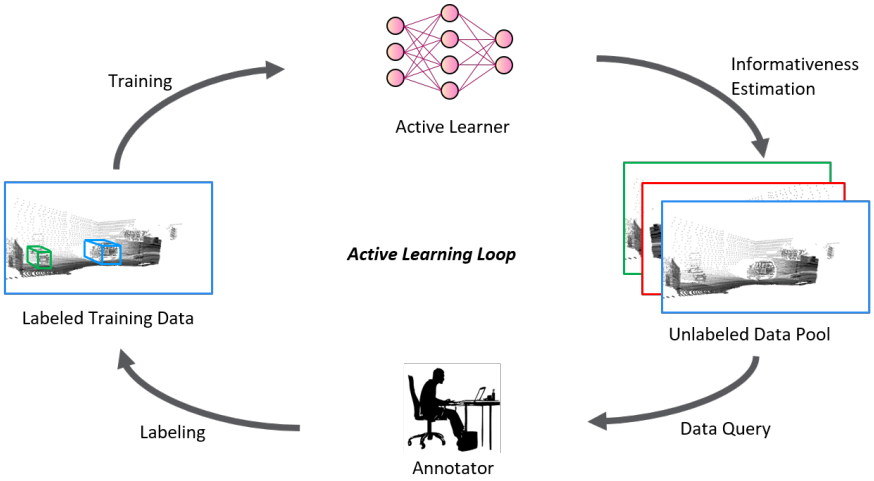


Figure 1.2: Pool-based active learning loop

For all scenarios of active learning, the core is how to estimate and query the most informative unlabeled samples for labeling. Many query strategies have been proposed in the literature. Uncertainty sampling [32] and query-by-committee (QBC) [50] are among the most popular ones. Uncertainty sampling queries instances that the learner feels most uncertain to label. Measurement of the uncertainty is often straightforward for probabilistic models. For example, with a binary Bayesian classifier, the uncertainty sampling strategy shall query the samples whose posterior probabilities of being positive are nearest 0.5[32]. Nevertheless, for some other machine learning algorithms, such as neural networks, the estimation of prediction uncertainty is still under discussion. Some propose to approximate the uncertainty for neural networks using Monte-Carlo (MC) inference [18] or ensemble methods [30]. Positive results are reported in various experiments [19][4]. Compared to uncertainty sampling, QBC query strategy has a stronger theoretical foundation. It queries in the controversial regions of the input space to minimize the version space which is the set of hypothesized models that are consistent with the current labeled training data. In other words, QBC queries on the disagreement among a committee of models which are trained on the same labeled dataset and each of them is able to vote for the labelings of query candidates. The samples that incite the most diverse prediction results from the com-

mittee are considered to be queried. Unfortunately, to deploy QBC in practice could consume large storage space, especially with a committee of deep neural networks. Due to limited time and computation resource, we concentrate on investigating active learning with uncertainty sampling in this thesis and leave the other query strategy frameworks as an interesting topic for future research.

1.2.2 3D Object Detection for Autonomous Driving

A 3D object detection task is essentially composed of two-subtasks: object classification and 3D bounding box regression (Fig.1.3). As is mentioned above, detection for autonomous driving is usually based on LiDAR and/or camera image data. LiDAR has the advantage of producing accurate depth information that is vital for 3D localization, and immune to illumination variation. Camera image suffers from too bright or dark light, but it provides abundant high-resolution texture information on the environment which is useful for classification.

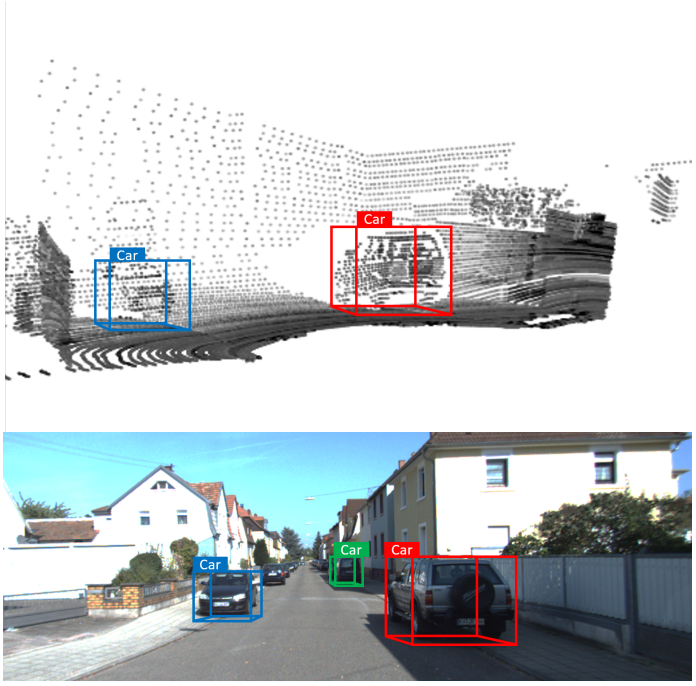


Figure 1.3: Illustrative example of 3D detection result with LiDAR point cloud (upper) and camera image (below). The detected object is denoted with a 3D bounding box and a class label.

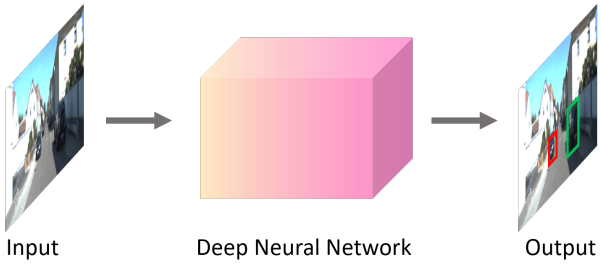
The state-of-the-art methods for object detection apply deep neural networks which can be classified into two categories, i.e. one-stage and two-stage detector. Given an input, a one-stage detector, such as YOLO [43] and SSD [36], directly outputs a certain number of bounding box and class predictions. On the contrary, a two-stage detector first explicitly generates region proposals or Region of Interest(ROI) which are candidate bounding boxes around potential objects, following by feature extraction, category classification, and fine-tuning of the proposal geometry [21]. Fig.1.4 demonstrates the main difference between these two detection pipelines. In general, one-stage detectors are faster but reach lower accuracy rates [43][36][38][53], while two-stage detectors are typically slower but able to reach the highest accuracy rates [44][39][3][53]. However, both of them utilize highly complicated backbone network, for instance, Residual Networks [22], to extract features from input at beginning of the pipeline.

Towards our goal of training a 3D object detector more economically, we endeavor to train a deep neural network detector under the framework of active learning. Henceforth, we abbreviate it as deep active learning for 3D detection (DAL3D). However, it is a challenging as well as innovative task, as there is a gap between our attempt and previous literature. The closest research to ours basically studies active learning for image classification or 2D image detection with emphasis on query strategy investigation and quantitative or qualitative analysis. To the best of our knowledge, studies on active learning framework with a 3D detector are rarely reported. In this thesis, we dive in DAL3D, design multiple stages of experiments and investigate deep active learning frameworks with 3D neural network detectors. In the following section, we will review related work on active learning and 3D object detection.

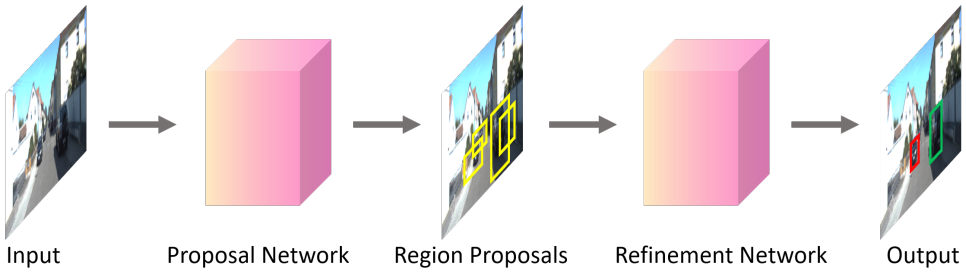
1.3 Related Work

1.3.1 Active Learning

There is a comprehensive survey of active learning [49]. In this section, we review related works on active learning for classification and object detection.



(a) One-stage detection pipeline



(b) Two-stage detection pipeline

Figure 1.4: (a) One-stage detector directly outputs bounding box regression and classification results. (b) Two-stage detector applies a proposal network to generate a bunch of region proposals first which are represented with yellow boxes. Then a refinement network (in some literature referred as "detection network") outputs final bounding boxes and classes. Note that some common operations such as prior anchor setting and non-maximum suppression are omitted in this graph for illustration convenience.

Active Learning for Classification

Plenty of research have studied active learning for classification with traditional classifiers, such as Support Vector Machine (SVM), Bayesian classifier, etc. Seyda et al. [15] combine small pool-based active learning with SVM to classify imbalanced data. Queries are made based on the distance of an instance to the support vector. Yi Yang et al. [62] add diversity maximization to an active multi-class SVM to avoid over-fitting on the small set of selected training data. Bayesian classifier is introduced to active learning using MC Dropout based entropy criteria as a representation of informativeness as in [46][41]. Moreover,

Qi et al. [41] exploit the correlations among labels to further improve data efficiency. The uncertainty sampling strategy has also been combined with other classifiers, such as the K-nearest neighbor classifier [35], Gaussian process [27], and the probabilistic K-nearest neighbor classifier [25].

Active learning has not been introduced with neural networks until recent years. One major obstacle is how to estimate the uncertainties in a neural network. Gal and Ghahramani [18] propose Monte Carlo (MC) dropout to formulate the model uncertainty of a Bayesian convolutional neural network for image classification. Gal's PhD dissertation [17] investigate various kinds of query functions based on MC Dropout in details and emphasizes the advantage of Monte Carlo active learning over passive learning and some semi-supervised learning methods. Dayoub, Sunderhauf, and Corke [11] also adopt MC Dropout neural networks and studies various updating strategies to improve active learning efficiency when data is obtained episodically. In contrast, Lakshminarayanan, Pritzel, and Blundell [30] propose to use an ensemble of networks to predict uncertainty. Beluch et al. [4] approximate the neural network uncertainty also using ensemble methods, and they constantly outperform MC uncertainty estimation in terms of data efficiency. They propose to explain the difference in performance by a combination of decreased model capacity (MC Dropout decreases model capacity because of dropout) and lower diversity (MC Dropout could be sampling from a locally optimal model posterior, but ensemble method has variational model initializations). Yan, Chaudhuri, and Javidi [61] propose a disagreement based active learning strategy with a classifier in general form on logged data. Multiple importance sampling and a debiasing query strategy are applied to work with a predetermined data logging policy.

Active Learning for Object Detection

There is very sparse literature on active learning for object detection. Most focus on 2D scenarios and few adopt neural network detector. To the best of our knowledge, we are the first to attempt for active learning in 3D object detection problem using deep neural networks.

Sivaraman and Trivedi [52] apply active learning with an Adaboost classifier in a 2D vehicle recognition and tracking system. The query strategy is fairly straightforward. The missed and false detection, along

with a few correctly identified vehicles are selected and added to the archived training data to avoid biased sampling. They further study three uncertainty based query strategies with two feature-classifier detectors (histogram of oriented gradient features and SVM classification, and Haar-like features and Adaboost classification) for 2D vehicle detection [51]. The discriminate classifiers are designed specifically to output probabilistic confidence on the detection result, which essentially transforms the active learning for 2D detection problem into a classification problem. Also, some propose version space-based active learning frameworks to train SVM based detectors on 2D objects of particular categories [54][47]. The samples that maximize version space reduction, i.e. whose addition to the SVM detector leads to minimum geometric margin, are chosen for annotation. Some extend active learning frameworks for video data in an incremental manner [55][63], in which they explicitly model the manual annotation cost of 2D objects. Vondrick and Ramanan [55] query human annotator for corrections on a full pass of tracking results, which limits the annotation efficiency by the tracker's speed. Yao et al. [63] stimulate the process by substituting the tracker with an incremental Hough Forests detector that gives live feedback on each input frame. Brust, Käding, and Denzler [6] study active learning with a deep convolutional neural network (YOLO-Small architecture [43]) as the 2D image detector. The unlabeled data are queried by classification uncertainty only. They also explore different ways of aggregating the query metrics of different objects from the same image frame and conclude that summing up metrics achieves the best overall performance. Roy, Unmesh, and Namboodiri [48] employ SSD [36] in active learning for 2D detection. The top extra convolutional layers, which output candidate bounding boxes that are close to each other, are viewed as a committee of classifiers. Therefore, the system queries based on the prediction score disagreement within the committee.

Nevertheless, the previous query strategies either resort to version space methods or classification uncertainty based selective sampling. The bounding box regression uncertainty is overlooked in query selection. In a recent paper [26], which studies active learning with two deep neural networks (Faster R-CNN [44] and SSD [36]) for 2D object detection, Kao et al. investigate various ways of incorporating 2D bounding box Localization Tightness (LT, i.e. the overlapping ratio between region proposal and final prediction) and Localization Stability

(LS, i.e. resilience of predicted location to noise) along with confidence based classification uncertainty for query. Among their results, query by LT with classification uncertainty yields the best data efficiency. In our work though, we explicitly model the bounding box regression uncertainty using prediction variance which can be interpreted as the variance of multiple regression results from the same region proposal. Details can be found in Chapter 2.

1.3.2 3D Object Detection

One-Stage Detection

Vote3Deep [14] proposes a computationally efficient approach for 3D object detection by exploiting the sparsity of LiDAR point clouds. Input clouds are discretized into a 3D grid, and each cell contains basic statistics extracted from the points within. Convolutions are only performed on non-empty cells while adopting a feature-centric voting scheme [56]. $\mathcal{L}_1$ regularization on the convolutional kernels and $ReLU$ activation are applied to further preserve the feature sparsity. However, the size of output bounding boxes is fixed for each class of objects based on train data statistics. LMNet [38] proposes a multi-class 3D LiDAR object detector with five layers of feature map (reflectance, range, three orthogonal views of LiDAR point clouds) as input. It achieves much faster real-time detection by utilizing dilated convolutions. Caltagirone et al. [7] project the 3D LiDAR point clouds into a 2D feature map for road surface detection. They essentially perform 2-class segmentation on the feature map using Fully Convolutional Network (FCN).

Contrast to crafting the input features, some explore end-to-end frameworks for 3D object detection. Following the idea of 2D end-to-end object detection, Li, Zhang, and Xia [34] project and discretize 3D LiDAR point clouds into a 2D point map which is directly fed to an FCN to predict 3D bounding box and objectiveness simultaneously for vehicle detection. Li [33] extends this work by processing 4D point cloud array brutally. However, the input dimension increment significantly consumes more computational resource. VoxelNet [64] removes manual feature representation by proposing Voxel Feature Encoding (VFE) layers that learn descriptive shape information from input point cloud voxels automatically. Through aggregating the local voxel features, an intermediate 3D convolutional network further transforms

the point cloud into a high-dimensional volumetric representation. Finally, a Region Proposal Network (RPN) [44] yields the detection result.

Some other works fuse LiDAR point clouds with images. Xu, Anguelov, and Jain [60] propose an early fusion strategy that directly learns to combine RGB image and LiDAR information. A convolutional network extracts appearance and geometric features from images and a variant of PointNet [40] processes raw LiDAR point cloud independently. A fusion sub-network aggregates the two outputs and predicts final 3D bounding boxes. Asvadi et al. [2] perform front-view 3D detection based on a late fusion of three modalities, including color image, up-sampled LiDAR depth map, and high-resolution LiDAR reflectance map. Each modality is processed by an individual YOLO network [43] to predict 3D bounding boxes. The three modalities are fused by a Multi-Layer Perceptron (MLP) network [58] which models the nonlinear mapping between the predicted bounding boxes and the ground truth.

Two-Stage Detection

Du, Ang, and Rus [12] fuse LiDAR and image data to generate region proposals. Seed 3D proposals are first synthesized from filtered LiDAR point clouds with hand-crafted heights. Small, medium and large bounding box proposals are respectively fused with extracted image features in RPNs to refine the 3D proposals. The proposals, as well as image features, are both fed to the last detection network which generates final bounding boxes and objectiveness scores. Chen et al. [10] propose MV3D, also a sensory-fusion framework, that encodes input LiDAR point clouds and RGB images in three views: bird's-eye-view (BEV) LiDAR, front view LiDAR, and image view. Given the BEV point clouds, 3D region proposals are generated from a set of 3D prior boxes and projected back to three views respectively to extract regional features for deep fusion which fuses multi-view features hierarchically using element-wise mean operation. However, MV3D fails to detect small objects effectively due to the down-sampled feature map. AVOD[29] raises a novel RPN architecture leveraging 1×1 convolutional filter to provide full resolution feature maps that help to improve recall and localization accuracy. AVOD also fuses image and LiDAR data, yet in two steps. Image and BEV LiDAR feature maps

are extracted and firstly fused with a 3D prior anchor grid to generate initial 3D proposals. The proposals are fused with the feature maps in the second step to output the final multi-class 3D detection result.

Matti, Ekenel, and Thiran [37] decouple 3D object localization and classification by processing LiDAR and image data separately. They first cluster the LiDAR point cloud using density based method to generate 3D bounding box proposals for target objects, i.e. pedestrians. The 3D proposals are refined by filtering on the size, adapting to the ground plane, and adjusting the aspect ratio to a fixed value. Then, they are projected onto 2D camera images to produce 2D candidate regions that are further processed by ResNet [22] based pedestrian classifier. Frustum Pointnet [39] proposes an opposite scheme that utilizes a 2D image detector first to generate frustum proposals in 3D LiDAR point clouds, and then applies 3D PointNets [40] to consecutively perform per point segmentation and amodal 3D bounding box regression within the frustum. The PointNet applies simple MLP and max pooing to preprocess input raw point clouds to make the model invariant to input permutation. They further concatenate extracted global features with input per point features to realize segmentation. Similarly, Du et al. [13] also apply image detection first to generate 2D proposals for vehicles and select a subset of point clouds. Then a model fitting algorithm finds the car points from the subset. Finally, instead of PointNet, a two-stage refinement convolutional neural network consumes the car points and predicts fine-tuned 3D bounding box and objectiveness score. Chen et al. [9] generate 3D bounding box proposals by minimizing a novel energy function on stereo images. The energy function incorporates object size priors, ground plane, point densities inside a box, visibility to the ground, and other depth informed features to improve proposal recall for occluded or small objects.

Feng, Rosenbaum, and Dietmayer [16] propose a probabilistic LiDAR 3D vehicle detection network that models the epistemic and aleatoric uncertainties. The epistemic uncertainty is extracted by performing Monte Carlo dropout during multiple forward passes, and the aleatoric uncertainty is captured using auxiliary output layers. They generate 3D proposals on BEV LiDAR feature map using RPN and finalize the classification and regression output simultaneously using MLP. Aleatoric uncertainty is incorporated in the total loss function during training to improve detection accuracy and robustness to noisy data. BirdNet [3] also performs 3D object detection using BEV LiDAR data

and RPN. However, training data statistics is required to generate the final 3D detection results.

In this work, we design multi-stage experiments and build lightweight 3D object detectors following the two-stage pipeline, as it allows us to design and implement experiments progressively. Compared to pursuing a highly accurate detector, our research interest rather lies in the study of how active learning behaves on improving training efficiency when incorporated with a deep neural network 3D detector. We first introduce the general architectures of the active learners for each stage experiment in Chapter 2, and discuss more in details in Chapter 3, 4, and 5.

1.4 Contribution and Organization of the Thesis

Motivated by the need for alleviating the training cost of a 3D object detector in autonomous driving, we propose DAL3D that leverages active learning framework to train a deep neural network 3D object detector. To the best of our knowledge, published literature at most study active learning with classification and 2D object detection. The leap to 3D detection remains challenging, as the design of query strategy and interactions between data, 3D active detector and the learning framework itself are not clear. We investigate the performance of deep active learning framework with multiple stages of experiments progressively. The first stage is a preliminary test of active learning for image classification to study some basic factors. In the second and third stage experiments, we develop and study two active learning frameworks for 3D detection mainly with LiDAR data. However, in the second stage, only the refinement network part of a two-stage detector is trained in learning loops. In the last stage, we train a complete two-stage 3D detector with active learning. Our main contributions in this thesis include:

- develop a deep active learning method to significantly reduce the labeling efforts for training a 3D object detector using LiDAR point clouds and accurate 2D image proposals
- investigate several uncertainty based sampling strategies in three

stages of experiments, and try to understand active learning behaviors by analyzing query data

- discuss unresolved challenges of deep active learning for 3D detection

In Chapter 2, we illustrates the deep active learning frameworks in three stages of experiments. The implementation details and experimental results are introduced and analyzed in Chapter 3, 4 and 5, respectively. Finally, we summarize the results and discuss some directions for future work in Chapter 6.

Chapter 2

Methodology

Towards the goal of reducing the training cost of a 3D detector with active learning, we design experiments in three stages that are increasingly complex but logically coherent:

(1) Deep Active Learning for Image Classification

(2) Deep Active Learning for Multi-Modal 3D Object Detection

(3) Deep Active Learning for LiDAR 3D Vehicle Detection

The pool-based active learning frameworks in these experiments share the same overall structure which has been shown in Fig.1.2. Also, the theoretical representation of prediction uncertainties and query criteria are essentially the same, though adaptations to different types of data, learner and query strategy are made accordingly. In this chapter, we briefly revisit the pool-based active learning framework, then illustrate the deep neural network learners that we build in different experiments. In the end, we discuss the core part which is uncertainty sampling query strategies.

2.1 Pool-Based Active Learning

In section 1.2.1, we have introduced the workflow of the pool-based active learning framework. Hereinafter, we refer to pool-based active learning simply as active learning, unless stated otherwise. It is virtually an iterative loop with four major components: active learner, unlabeled data pool, annotator, and labeled training data. When the

framework incorporates different sorts of learners, we adjust the informativeness measurement and query criteria accordingly, so that appropriate unlabeled samples could be selected from the pool and annotated. For example, when we have a classifier as the learner, we need to measure the classification uncertainty and based on which make the queries. Or in other cases, we have a detector that outputs both classification and regression predictions, then both types of uncertainties shall be considered, and multiple query criteria exist to exploit the uncertainties.

We implement our methods based on open source datasets, where each sample is provided with a ground truth label. Before query, an active learner is isolated from labels and only has access to inputs from the unlabeled data pool. After query, we annotate the selected samples by retrieving their labels, and then update them to the training set.

2.2 Active Learners

We introduce the active learners in our three stages of experiments.

First Stage Experiment

We build a simple image classifier (Fig.2.1) with convolutional layers to extract features and fully connected layers with softmax output to perform classification. At this stage, one input is a single frame of image containing a single object to be classified. Queries are made solely based on the estimated classification uncertainty of the learner.

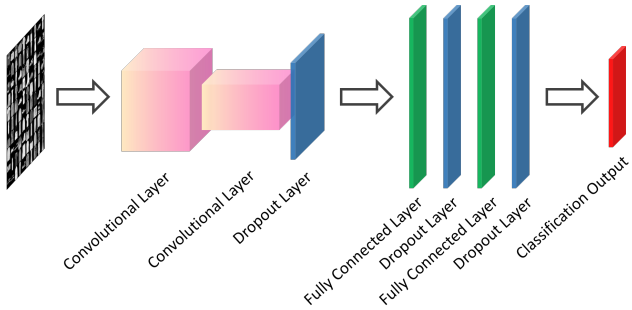


Figure 2.1: Deep active image classifier. Inputs are images with a single object.

Second Stage Experiment

Now, we move a step further from classification to 3D detection and build the detector following two-stage R-CNN pipeline. However, the active learner that we are interested in at this stage is not the entire detector, but its refinement network that extracts higher level features from ROIs and performs classification and bounding box regression to output the final detection results (Fig.2.2). Such choice of active learner benefits our progressive research in two-folds. First, large low-level feature extraction backbones are not involved into the active learning loop, which reduces runtime and facilitates our experiments. Second, when the active learner deals with ROIs that each contains at most one object, we do not need to consider the complicated combined informativeness measurement of multiple objects from a single input, but focus on measuring the informativeness of every single object, which is similar to what we do in the previous stage of experiment.

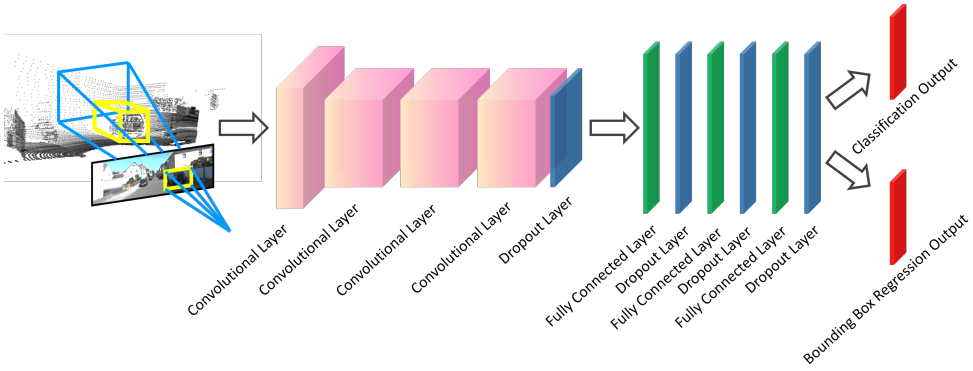


Figure 2.2: Deep active 3D multi-modal detector. Inputs are multi-modal region proposals generated from 3D LiDAR point cloud frustums that lie within 2D image ROIs.

The inputs of the active learner, i.e. ROIs, are generated based on multi-modalities. We first synthesize seed 2D region proposals using camera images and an on-the-shelf well-trained 2D object detector (such as YOLOv3[42], Faster R-CNN[44], etc.). Then, the LiDAR frustum that projected within a 2D proposal is selected as a 3D proposal. Note that the proposal is 3D not because the data are structured as a 3D point cloud array, but it contains 3D depth information. Finally, 2-channel feature maps that incorporate the depth and LiDAR intensity information of the 3D proposals are fed to the active learner as inputs.

Third Stage Experiment

Last but not least, we study active learning with a complete two-stage 3D LiDAR detector(Fig.2.3), whose architecture follows a Faster R-CNN pipeline[16]. Input LiDAR point clouds are first pre-processed into BEV multi-layer feature maps. A backbone network extracts comprehensive features for an RPN[44] to generate a constant number of BEV 3D proposals on the point cloud feature maps. Multiple fully connected layers refine the proposals and predict the final 3D detection results.

In the previous two stages, the learners essentially take objects as inputs and query on individual objects. However, in the third stage the network takes frames of LiDAR point clouds as inputs and queries also by frames, i.e. selects complete frames of point clouds as queries, regardless of how many objects are included in one frame. The informativeness of each input is approximated by aggregating the prediction uncertainties of all ROIs on it.

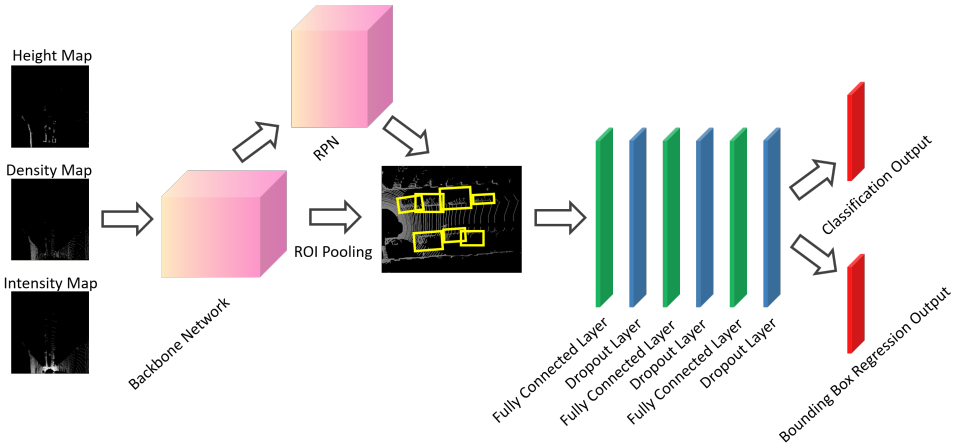


Figure 2.3: Deep active 3D LiDAR detector. Inputs are pre-processed BEV LiDAR feature maps.

2.3 Query Strategies

A query strategy for active learning determines how to measure and select the most informative samples from the unlabeled data pool. As mentioned in Sec.1.2.1, we concentrate on uncertainty sampling strat-

egy that evaluates the prediction uncertainty on each unlabeled sample and utilizes an acquisition function to select the most uncertain ones based on some mathematical criteria. In the following context, we introduce how to estimate the prediction uncertainty for deep neural networks and various kinds of acquisition functions. However, it is worth mentioning that the uncertainty estimation can be more complicated when a single input corresponds to multiple prediction results. For instance, a single frame of LiDAR point cloud sample containing several objects yields multiple bounding box and classification predictions. We need to deal with this problem in the last stage experiments. In Chapter 5, adaptations are made based on the general strategies below.

2.3.1 Uncertainty Estimation for Deep Artificial Neural Networks

As for 3D detection, there are two sources of prediction uncertainty, i.e. classification uncertainty and bounding box regression uncertainty.

Classification Uncertainty

We estimate classification uncertainty by the predictive probability $p(y = c|x)$. x denotes an input sample, $y = c$ denotes prediction y as class c. A direct way to obtain predictive probability is by softmax output, i.e. $p(y|x) = softmax(x)$. However, as discussed in several works (e.g. [17][4]), the softmax output may assign high probability to unseen data, resulting in over-confident predictions. Therefore, we also introduce two other approaches that have been studied recently to estimate the prediction uncertainty for deep neural networks, namely, *MC Dropout*[17] and *Deep Ensembles*[30].

The idea behind *MC Dropout* is to approximate the predictive probability under the Bayesian neural network framework. Previously, Bayesian neural network is hard to implement due to the difficulty of inferring the model posterior when having a large number of parameters. However, [17][18] prove that by applying dropout in multiple forward passes, i.e. inferences, model posterior can be efficiently approximated without introducing extra parameters, and predictive probability can be represented by the average of multiple dropout inferences. More specifically, given an input sample x, the neural net-

work performs T times forward passes with dropout. The classification uncertainty is approximated by the averaged softmax output:

$$\begin{aligned} p(y=c|\mathbf{x}) &\approx \frac{1}{T}\sum_{t=1}^{T} p(y=c|\mathbf{x}, \hat{\mathbf{w}}_\mathbf{t}) \\ &= \frac{1}{T}\sum_{t=1}^{T} softmax_{(\hat{\mathbf{w}}_\mathbf{t})}(\mathbf{x}) \end{aligned} \tag{2.1}$$

in which, $\hat{\mathbf{w}}_\mathbf{t}$ stands for the network's weights in i-th inference.

Compared to *MC Dropout*, *Deep Ensembles* estimates prediction uncertainty in a non-Bayesian way[30]. It suggests to train several networks with the same architecture but random initialization, and average the networks' softmax outputs. Despite that *Deep Ensembles* method deactivates dropout during prediction, the estimated classification uncertainty still shares a similar form with *MC Dropout*. Let E be the number of ensembles and $\mathbf{W_e}$ a single network in the ensemble, we have:

$$\begin{aligned} p(y=c|\mathbf{x}) &\approx \frac{1}{E}\sum_{e=1}^{E} p(y=c|\mathbf{x}, \mathbf{W_e}) \\ &= \frac{1}{E}\sum_{e=1}^{M} softmax_{(\mathbf{W_e})}(\mathbf{x}) \end{aligned} \tag{2.2}$$

Regression Uncertainty

As for a single input $\mathbf{x}$, the regression uncertainty of a predicted bounding box can be characterized by its total variance TV from multiple inferences[49][16]. Let us denote the predicted bounding box from i-th inference as a vector $\mathbf{v_i}, i \in \{1, 2, ..., T\}$. The mean is:

$$\overline{\mathbf{v}} = \frac{1}{T}\sum_{i=1}^{T} \mathbf{v_i} \tag{2.3}$$

And its covariance matrix is:

$$\mathbf{Cov}(\mathbf{x}) = \frac{1}{T}\sum_{i=1}^{T} \mathbf{v_i}\mathbf{v_i^T} - \overline{\mathbf{v}}\overline{\mathbf{v}}^\mathbf{T} \tag{2.4}$$

Then we have the total variance as the trace of the covariance matrix:

$$TV(\mathbf{x}) = trace(\mathbf{Cov}(\mathbf{x})) \tag{2.5}$$

The larger TV, the more uncertain is the prediction.

2.3.2 Acquisition Function

In general, given a model $\mathcal{M}$, unlabeled data pool $\mathcal{D}_{pool}$, and inputs $\mathbf{x} \in \mathcal{D}_{pool}$, an acquisition function $f(\mathbf{x}, \mathcal{M})$ is a function of $\mathbf{x}$ that the active learning system uses to decide where to query:

$$\mathbf{x}^* = \underset{\mathbf{x} \in \mathcal{D}_{pool}}{\arg\max}\, f(\mathbf{x}, \mathcal{M}) \tag{2.6}$$

With the predictive probability of classification $p(y = c|\mathbf{x})$ and total variance of bounding box regression TV defined above, we are able to formulate and calculate the acquisition function $f(\mathbf{x}, \mathcal{M})$ more specifically.

Acquisition Functions for Classification

A straightforward measurement of uncertainty is Shannon Entropy (SE). One option is to query samples that maximizes the predictive entropy (*Maximize Entropy*):

$$\mathcal{H}[y|x] := -\sum_{c}^{C} p(y = c|\mathbf{x}) \log p(y = c|\mathbf{x}) \tag{2.7}$$

where C is the total number of classes.

In addition, since both *MC Dropout* and *Deep Ensembles* offers samples from the predictive probability distribution (i.e. $p(y = c|\mathbf{x}, \hat{\mathbf{w}}_\mathbf{t})$ and $p(y = c|\mathbf{x}, \mathbf{W_e})$), we can utilize them to query those samples that maximize the information gained about the model parameters (*Maximize Information Gain*, or *BALD*[19]):

$$\mathcal{I}[y, \mathbf{w}|\mathbf{x}] := \mathcal{H}[y|\mathbf{x}] - E_{p(\mathbf{w}|\mathcal{D}_{train})}[\mathcal{H}(y|\mathbf{x}, \mathbf{w})] \tag{2.8}$$

In which, $\mathbf{w}$ is the model parameters, $\mathcal{H}(y|\mathbf{x}, \mathbf{w})$ is the entropy of y given $\mathbf{w}$, $p(\mathbf{w}|\mathcal{D}_{train})$ indicates the posterior probability distribution of $\mathbf{w}$. When using *MC Dropout*, we further rewrite Eq.2.8 as:

$$\mathcal{I}[y, \mathbf{w}|\mathbf{x}] \approx \mathcal{H}[y|\mathbf{x}] + \frac{1}{T}\sum_{t=1}^{T}\sum_{c=1}^{C} p(y=c|\mathbf{x}, \hat{\mathbf{w}}_\mathbf{t}) \log p(y=c|\mathbf{x}, \hat{\mathbf{w}}_\mathbf{t}) \quad (2.9)$$

As for *Deep Ensembles*, we just need to replace the $\hat{\mathbf{w}}_\mathbf{t}$ with $\mathbf{W}_\mathbf{e}$ and t with e in Eq.2.9.

Acquisition Function for Regression

Compared to classification, it is simpler to formulate the regression acquisition function that is directly based on TV. The learner shall query samples that yields the largest TV.

$$\mathbf{x}^* = \underset{\mathbf{x}\in\mathcal{D}_{pool}}{\arg\max}\, TV(\mathbf{x}, \mathcal{M}) \quad (2.10)$$

In which, TV is calculated via Eq.2.5.

Baseline Acquisition Function

The acquisition functions above are compared with a baseline method that randomly selects query data from the unlabeled data pool for annotation(Eq.2.11). In other words, the baseline acquisition function mimics the traditional passive learning pipeline that consumes training data without selection.

$$\mathbf{x}^* = \underset{\mathbf{x}\in\mathcal{D}_{pool}}{\arg\max}\, Random(\mathbf{x}) \quad (2.11)$$

Chapter 3

Deep Active Learning for Image Classification

As a preliminary test, in this chapter we first build an deep active learning framework for image classification on a synthetic dataset. We investigate some decisive factors of the proposed framework and conclude guidelines for the latter experiments.

3.1 Implementation Details

3.1.1 Data

All experiments in this chapter are performed on the Fashion MNIST dataset[59]. The Fashion MNIST resembles the MNIST dataset[31] in terms of data format and quantity, yet it contains more complicated imagery contexts. Exemplary Fashion MNIST samples are shown in Fig.3.1. There are in total 10 categories of objects. Each input is a 28×28 image containing one object.

By default, the Fashion MNIST dataset is partitioned with 10000 samples for testing and 60000 samples for training. They are evenly distributed among classes. In our experiments, we randomly select 10000 samples from the default training set for validation and less than 1000 samples as the initial training set to initialize the model before entering the active learning loop. To prevent bias, the initial training dataset and validation set are chosen with balanced labels. The rest of the default training set constitutes the unlabeled data pool.

Figure 3.1: Samples from the Fashion MNIST dataset

3.1.2 Active Classifier

All experiments at this stage are assessed with the same classifier architecture (Fig.3.2): convolution-relu-convolution-relu-max pooling-dropout-dense-relu-dropout-dense-dropout-softmax, with 32 convolution kernels, 4x4 kernel size, 2x2 pooling, 128 units in hidden layers, and dropout probabilities 0.25, 0.5 and 0.5 (based on the example Keras MNIST CNN implementation[28]). The classifier is trained with Adam optimizer and cross-entropy loss for 20 epochs in a single loop.

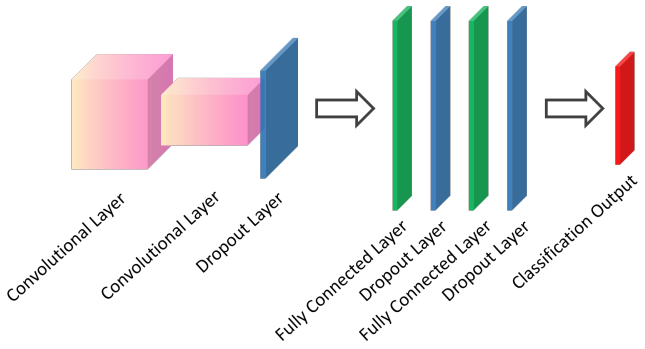


Figure 3.2: Architecture of the active classifier

3.1.3 Learning Loop Formulation

The workflow of an active learning loop and core components have been introduced in Chapter 2. In this chapter, we follow the same pattern to formulate the loop but substitute the learner with a neural network classifier. The updated learning loop and algorithm are shown in Fig.3.3 and Alg.1, respectively. The framework takes raw images from the dataset as input. The initial model and the updated models from every active learning iteration are saved. The loop stops after reaching a pre-determined number of iterations and all saved models are evaluated on the test set.

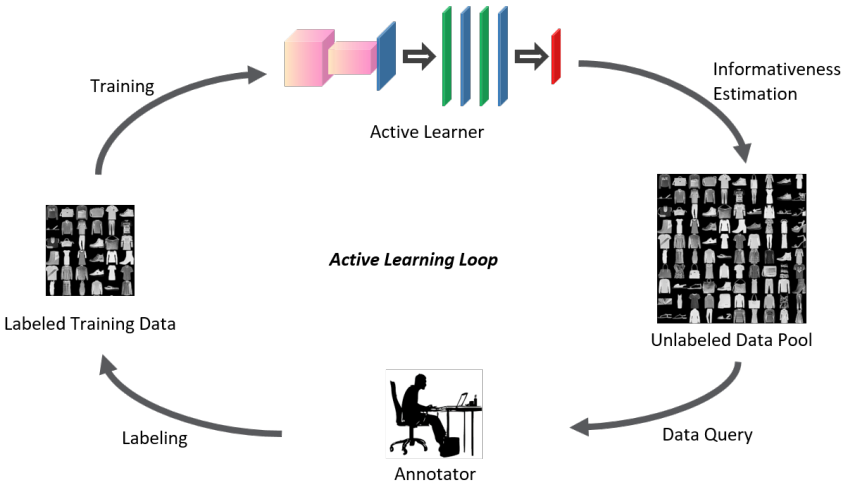


Figure 3.3: Active learning loop for classification

3.2 Experiments and Results

In order to gain some insights on active learning framework, the preliminary experiments evaluate four principle active learning factors on Fashion MNIST: size of query step, model initialization, choice of acquisition function, and model update strategy. The terms are defined respectively in the following subsections and each factor is investigated individually with the others held constant.

In all experiments of this chapter, the classification uncertainties are estimated using *MC Dropout* with 20 times forward passes. The active learning curve shown in the figures below are averaged over 20 repetitions. Y-axis denotes the classification accuracy on the test

Algorithm 1: Deep Active Learning for Image Classification

Input: Initial training dataset X_{train}, unlabeled pool X_{pool}, human/machine annotator A
Result: Classifier M

```
RandomInitialize M
M = TrainNetwork(X_train, M)
SaveModel M
while not StopCondition do
    U = ∅
    for x in X_pool do
        u_cls = EstClsUncertainty(x, M)
        ▷ Estimate classification uncertainty
        U.append(u_cls)
    end
    X* = AcquisitionFcn(X_pool, U)      ▷ Query the most informative samples
    Y* = Label(X*, A)
    X_train = X_train ∪ {X*, Y*}        ▷ Update training dataset
    X_pool = X_pool \ X*                ▷ Update unlabeled pool
    M = ReTrainNetwork(X_train, M)      ▷ Retrain classifier
    SaveModel M
end
```

set and X-axis denotes the total number of utilized labeled data, i.e. training data. Note that training dataset is supplemented constantly with queried data.

3.2.1 Size of Query Step

The size of query step is defined as the number of data that are selected by the acquisition function from the unlabeled data pool and queried for labeling in a single active learning iteration. Testing with different sizes of query step ($\in \{50, 100\}$) and fixed size of initial training dataset

(=1000), Figure 3.4 shows that the step size has no significant impact on the active learning curve. The model performs similarly given the same absolute number of labeled data.

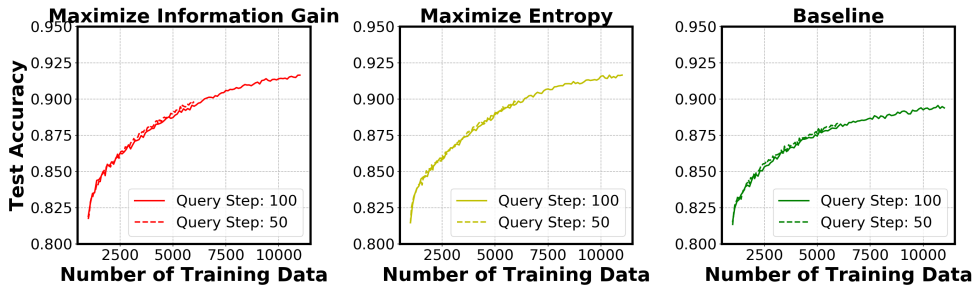


Figure 3.4: Different query step sizes with different acquisition functions

3.2.2 Model Initialization

We initialize the model with a small sub-set of labeled data. They are randomly selected but balanced in terms of classes. We test small and large number of initial training data (100 or 1000 samples) respectively, while the query step size is fixed as 100 per iteration. Figure 3.5 shows that more initial training data helps to improve the performance of active in the starting phase. Initialized with less training data, the model fails to develop a relatively accurate description on the prediction uncertainty and therefore the queried data are not so helpful to improve the model prediction accuracy compared to the model initialized with larger dataset.

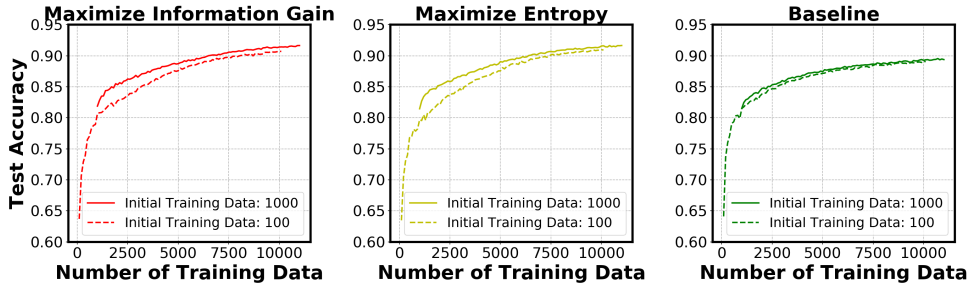


Figure 3.5: Different model initializations with different acquisition functions

3.2.3 Acquisition Function

We investigate the classification acquisition functions introduced in the previous chapter, i.e. *Maximize Entropy*, *Maximize Information Gain(BALD)*, and baseline. Figure 3.6 shows that given the same number of labeled data, *Maximize Entropy* performs similarly to *BALD*, and both of them outperforms baseline method, indicating that active learning is potentially a viable solution to reduce the need of labeled data.

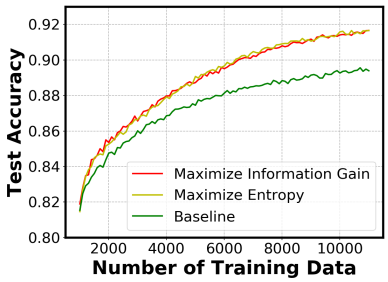


Figure 3.6: Different acquisition functions

3.2.4 Model Update Strategy

The model update strategy refers to how to utilize acquired labeled query data to update the learner. In Alg.1, we simpley use $ReTrainNetwork$ to denote the update step. Here is more detailed discussion. We use M_t to denote the model after t_{th} active learning iteration, A_i to denote queried data from $i-$th iteration, $i \in 1, 2, ..., t$, and $\otimes$ to denote the operation "train on". Two popular model update strategies[11] can be represented as:

- Traditional active learning

$$M_t = M_0 \otimes \bigcup_{i=1}^{t} A_i$$

- Incremental active learning

$$M_t = M_{t-1} \otimes \bigcup_{i=1}^{t} A_i$$

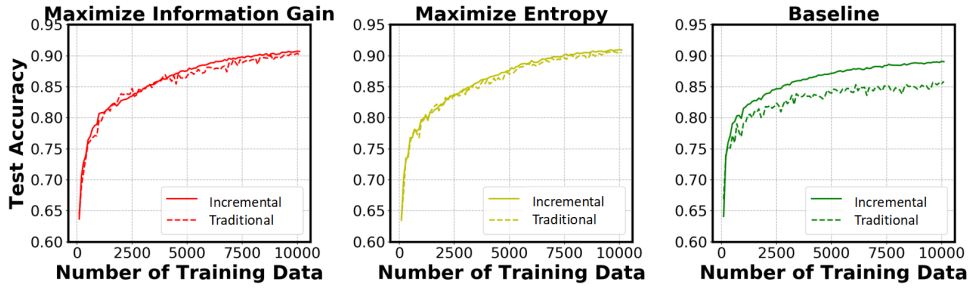


Figure 3.7: Different model update strategies

Figure 3.7 shows that, when using an effective acquisition function such as Maximize Entropy or maximize information gain, incremental active learning improves the test accuracy slightly (1%) given the same number of labeled data. Also, it improves the prediction performance significantly (3%) for the baseline acquisition function. One reason of the improvements yielded by incremental model update strategy is that re-using the model from the previous iteration essentially prolongs the number of training epochs. Or in other words, incremental active learning exploits the given labeled data more than traditional active learning does and hence yields better prediction performance with the same number of training data. However, this also increases the risk of over-fitting. In the experiments hereafter, we adopt traditional update strategy to avoid over-fitting, and isolate the benefits of active learning framework from merely prolonging training process.

3.3 Short Summary

In this chapter, we studied single-task active learning with a classifier on the Fashion MNIST dataset. To reach 89% test accuracy, active learning saves about 45.5% labeled data compared to the random sampling baseline. Through experiments, we found that model initialization plays an important role in active learning, as it helps the model to develop better uncertainty estimation. The size of query step exerts trivial influence on the results. The two acquisition functions (i.e. *BALD* and *Maximize Entropy*) shows small differences on the learning curves. However, as is pointed out in [49], the best query strategies can vary from case to case. Therefore the choice of uncertainty estimation methods and acquisition functions are still worth investigating in

the following experiments.

Chapter 4

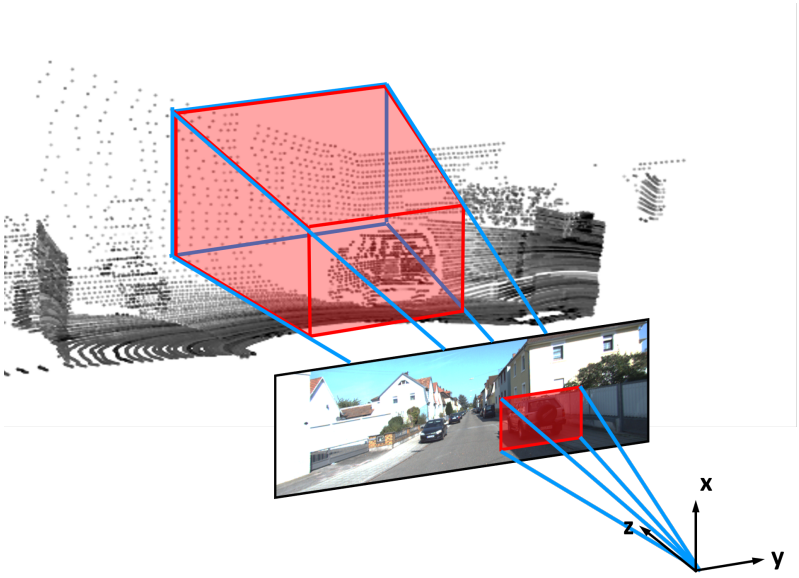
Deep Active Learning for Multi-Modal 3D Object Detection

In Chapter 3, we implement preliminary tests with an active classifier to investigate some primary factors of single-task active learning where the learner has only one type of output, i.e. classification. In this chapter, we study a more challenging multi-task active learning framework that incorporates a multi-modal 3D detector. The detector follows the two-stage R-CNN[21] pipeline, while the active learning framework only updates its refinement network which consumes multi-modality 3D proposals to predict final 3D bounding boxes and classes. As is mentioned in Chapter 2, such simplification on the active learner speeds up iteration and grants us the convenience of studying query strategy with single object as input, which is similar to our preliminary tests. In the next stage experiment, i.e. Chapter 5, we will explore deep active learning further with a full 3D LiDAR vehicle detector.

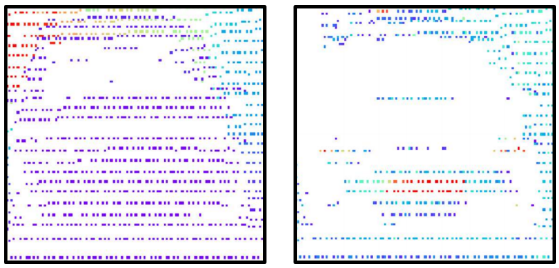
4.1 Implementation Details

4.1.1 Data

We create a dataset that consists of multi-modal sensory information as input and 3D bounding boxes and classes as ground truth based on the calibrated KITTI[20] image and LiDAR data. Still, we keep the pipeline of producing such dataset as general as possible. First, we project a single frame LiDAR point clouds onto the corresponding



(a) LiDAR frustum proposals



(b) Exemplary feature map (left: LiDAR depth map, right: LiDAR intensity map)

Figure 4.1: Generation of multi-modal feature map for the active learner. We obtain frustum candidates from 2D image region proposals. The 3D frustums containing depth and LiDAR intensity information are later projected into feature maps of two channels in the image frame.

camera image frame. Each point is associated with a depth value of its distance to the LiDAR. The distance is defined as $d = \sqrt{y^2 + z^2}$ in the LiDAR coordinate, with y pointing rightwards and z pointing forwards, as defined in Fig.4.1a. Second, we utilize 2D region proposals on the camera image frame to crop the projected point clouds accordingly so that LiDAR frustum proposals containing 3D depth information are obtained. Note that to avoid the influence from the RGB image

detector that produce the 2D proposals, we assume a "perfect" image detector that provides only accurate object proposals. This is achieved by extracting objects using their ground-truth labels with random margins. Third, we stacked up and resize the depth and LiDAR intensity maps of the proposed LiDAR frustums to formulate the final sparse feature map as input ($100 \times 100 \times 2$). No interpolation is performed to avoid artifacts. The projection of LiDAR point clouds and generation of frustum proposals are illustrated in Fig.4.1a. An exemplary multi-modality feature map is shown in Fig.4.1b. Every input sample has a corresponding class and a 3D rectangular bounding box which is denoted with its width, height, length, and centroid distance to the LiDAR. To increase numerical stability while training, we normalize the labels with constant scaling factors which are determined based on the dataset heuristics.

There are five classes in our customized dataset: small vehicle, human, truck, tram, and miscellaneous. Data distribution over classes is shown in Fig.4.2. Despite the underlying data unbalance, we randomly choose 200 samples from each class to compose the initial training set. The validation and test set consist of 3000 and 6000 randomly selected samples, respectively. As our preliminary tests show that the query step size has a trivial influence on the active learning performance, we fix the query size of 200 samples in our experiments.

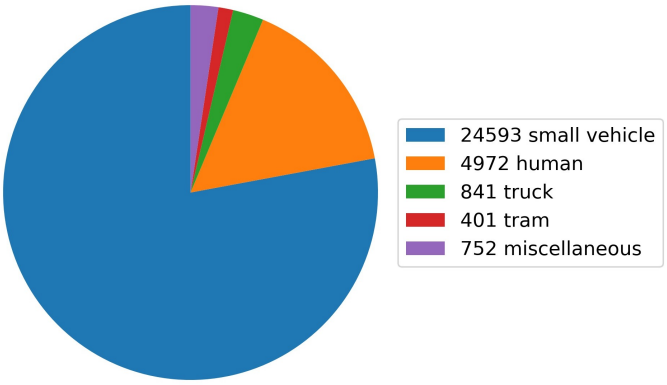


Figure 4.2: Data distribution over classes

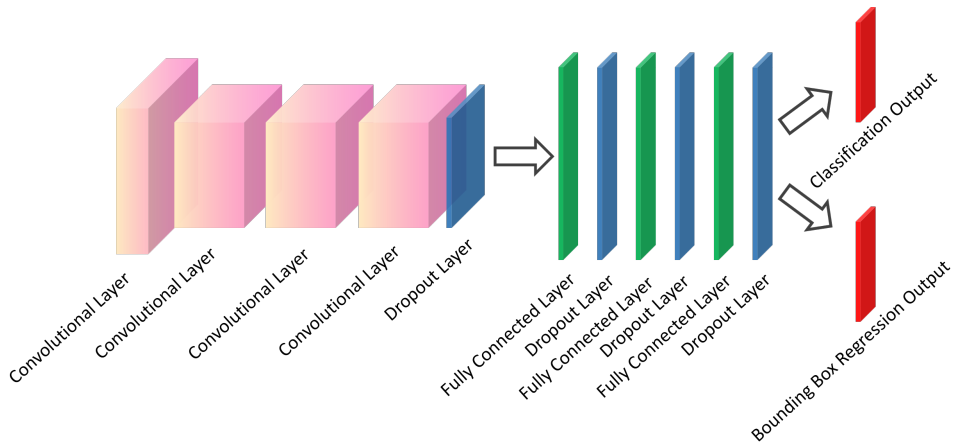


Figure 4.3: Architecture of the active multi-modal 3D detector

4.1.2 Active Refinement Network

We assume a two-stage R-CNN detector, in which the architecture of the refinement network is shown in Fig.4.3. Four convolution layers (each with 32 3×3 kernels and relu activation) and one dropout layer with a dropout rate of 0.5 extract features from the region proposal feature map, followed by three fully connected layers (each with 256 hidden neurons and a dropout rate also of 0.5). The 3D bounding box regression head and classification head run in parallel at the end.

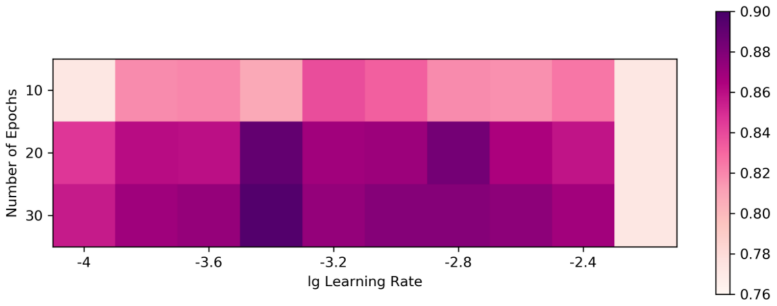


Figure 4.4: Classification accuracy heatmap for the search of appropriate learning rate and number of training epochs

The network is trained also with Adam optimizer. The loss function is defined as the linear sum of cross-entropy loss from the classification head and mean-square-error (MSE) loss from the regression head. The performance of the model is evaluated by both classifica-

tion accuracy and bounding box regression MSE. We search the initial learning rate and the number of epochs in each learning loop based on classification accuracy, which is shown in Fig.4.4.

4.1.3 Learning Loop Formulation

The active learning loop is shown in Fig.4.5. The refinement network of a two-stage 3D object detector acts as the active learner. It estimates both classification and regression uncertainties to measure the informativeness of an input sample. As is introduced in Chapter 2, we estimate the classification uncertainty by predictive probability with either *MC Dropout* or *Deep Ensembles*, and the regression uncertainty by *Total Variance*. The acquisition function queries either on one of the two uncertainties or the linear sum of both. Alg.2 illustrates the proposed multi-modal active learning pipeline for 3D detection.

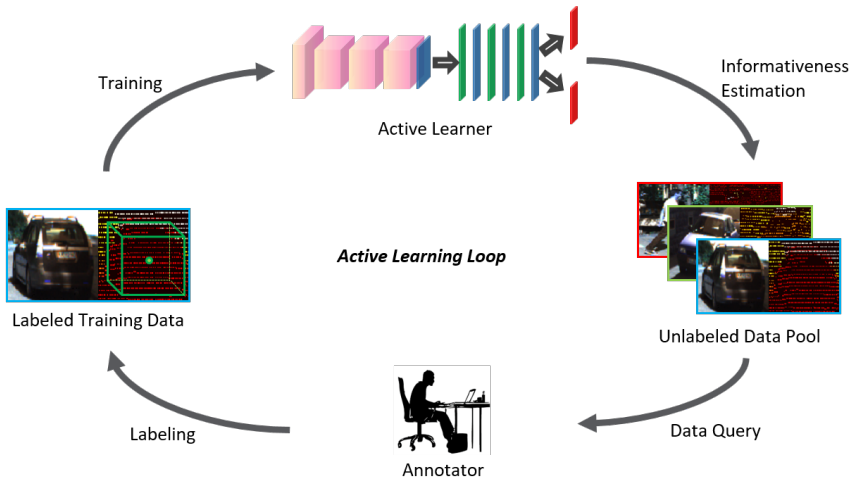


Figure 4.5: Active learning loop for multi-modal 3D detection

4.2 Experiments and Results

A major question for multi-task active learning is how to design a query criterion that balances multiple sources of uncertainty. Moreover, whether the classification uncertainty estimation methods (i.e. *MC Dropout* and *Deep Ensembles*) and the entropy-based acquisition functions (i.e. *BALD* and *Maximize Entropy*) are able to characterize

Algorithm 2: Deep Active Learning for Multi-Model Object Detection

Input: Initial training dataset X_{train}, unlabeled pool X_{pool}, human/machine annotator A
Result: Refinement network M
RandomInitialize M
$M = TrainNetwork(X_{train}, M)$
$SaveModelM$ **while** *not StopCondition* **do**
  $U = \emptyset$
  for x **in** X_{pool} **do**
    $u_{cls} = EstClsUncertainty(x, M)$
    ▷ Estimate classification uncertainty
    $u_{reg} = EstRegUncertainty(x, M)$
    ▷ Estimate regression uncertainty
    $U.append([u_{cls}, u_{reg}])$
  end
  $X^* = AcquisitionFcn(X_{pool}, U)$ ▷ Query the most informative samples
  $Y^* = Label(X^*, A)$
  $X_{train} = X_{train} \cup \{X^*, Y^*\}$ ▷ Update training dataset
  $X_{pool} = X_{pool} \setminus X^*$ ▷ Update unlabeled pool
  $M = ReTrainNetwork(X_{train}, M)$ ▷ Retrain classifier
  $SaveModel\ M$
end

data informativeness in our multi-modal 3D detection task is unclear. In the following sections, we introduce our experiments on different combinations of query criteria, the performance of uncertainty estimation methods and acquisition functions, and interpret the results based on query analysis. The classification accuracy and bounding box regression MSE are shown separately. *MC Dropout* applies 20 times forwards passes, *Deep Ensembles* adopts 5 individual models, and the *Total*

Variance is calculated from 10 times outputs. All results are averaged from 3 runs, and variances are also presented on the curves.

4.2.1 Query Criteria

We study four types of query criteria: query by maximal classification uncertainty (*MC Dropout + BALD*) or regression uncertainty (*Total Variance*) only, linear sum of both, or random sampling (baseline). Note that classification and regression uncertainty are usually of different magnitudes, we use the weighted sum to prevent bias. The results are shown in Fig.4.6.

In terms of classification accuracy, all active learning methods eventually outperform baseline. When querying only by the classification uncertainty, the learning curve quickly gains classification accuracy and out-performs the baseline with a large margin. However, when querying by only regression uncertainty or the summed uncertainties, the learning curve grows significantly slower. This indicates that *MC Dropout + BALD* is able to select critical samples reliably with an immature model, while querying by *TV* requires a better trained model to develop good representation of the sample informativeness. In other words, querying by regression uncertainty needs a better-initialized model. As for regression accuracy, querying by *MC BALD* out-performs the others marginally. On both evaluation metrics, results from querying by summed uncertainties are very close to querying by *TV* only, which indicates that regression uncertainty is more dominant on sample selection. In the following experiments, we focus on query by only classification uncertainty, as it displays better adaption to our active learner.

4.2.2 Uncertainty Estimation

Fig.4.7 shows the learning curves with *MC Dropout* and *Deep Ensembles* for uncertainty estimation. Given the same acquisition function, they have similar performances and both out-run baseline method. However, *Deep Ensembles + BALD* is slightly better than *MC Dropout + BALD* on classification with a relatively small number of training data (less than 7000).

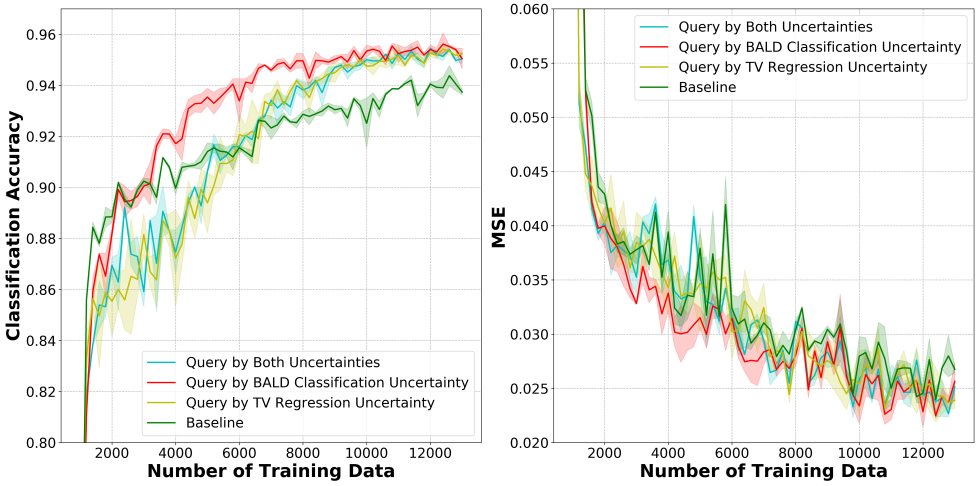


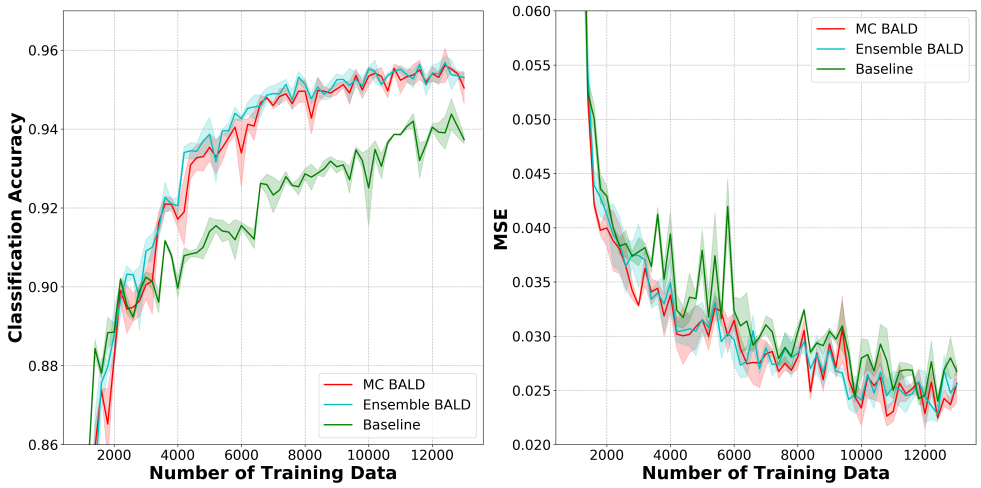
Figure 4.6: Active learning curves with four query criteria. X-axis denotes the increasing number of labeled data. Y-axis denotes the classification accuracy (left) or regression MSE (right).

4.2.3 Acquisition Function

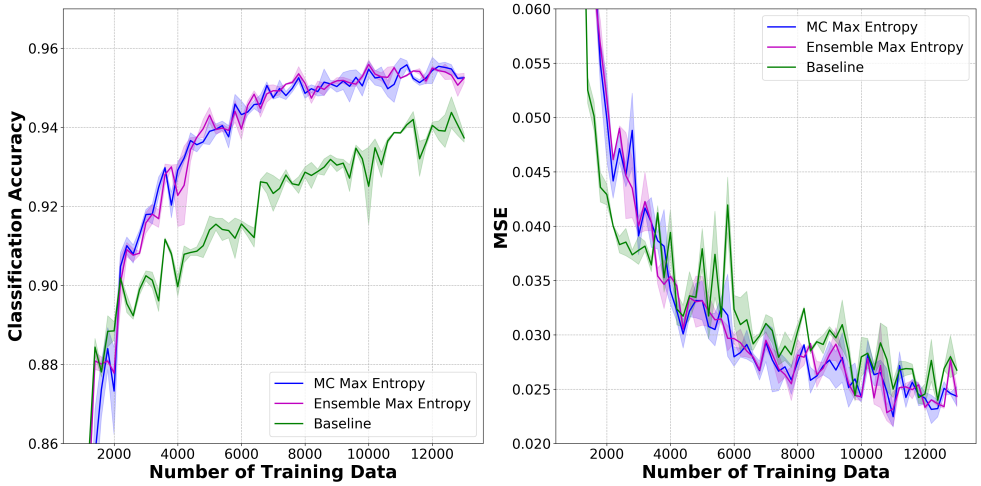
Moreover, we study different acquisition functions, i.e. *BALD* and *Maximize Entropy*. Fig.4.8 shows that when the amount of labeled training data is low (e.g. <4000), *BALD* outperforms *Maximize Entropy* on regression MSE in either way of estimating classification uncertainty, whereas *Maximize Entropy* achieves better classification accuracy with MC Dropout.

4.2.4 Annotation Cost Savings

Table 4.1 compares the number of labeled training data required to train the detector to reach a certain level of error relative to the full-trained model. Note that the latter achieves 95.67% classification accuracy and 0.02361 regression MSE, which is considered as the performance upper bound. The relative classification error is defined as the difference between the classification accuracy of the full-trained model and an active learning model. The relative regression MSE error is defined similarly, but with an extra division by the full-trained model MSE. The table also shows the percentage of labeling efforts saved by active learning methods in comparison to baseline. Our active learner reaches the same relative error with significantly less training data,



(a) *MC Dropout*/*Deep Ensembles* for uncertainty estimation with *BALD*



(b) *MC Dropout*/*Deep Ensembles* for uncertainty estimation with *Maximize Entropy*

Figure 4.7: Active learning curves by estimating classification uncertainty with *MC Dropout* or *Deep Ensembles*. The acquisition function is fixed as *BALD* and *Maximize Entropy* subsequently

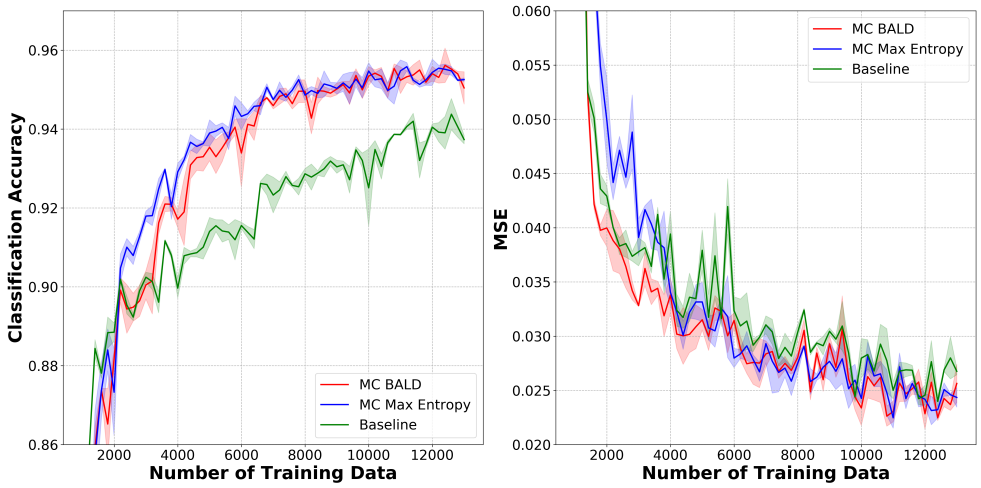
saving up to 60% labeling efforts.

4.2.5 Query Analysis

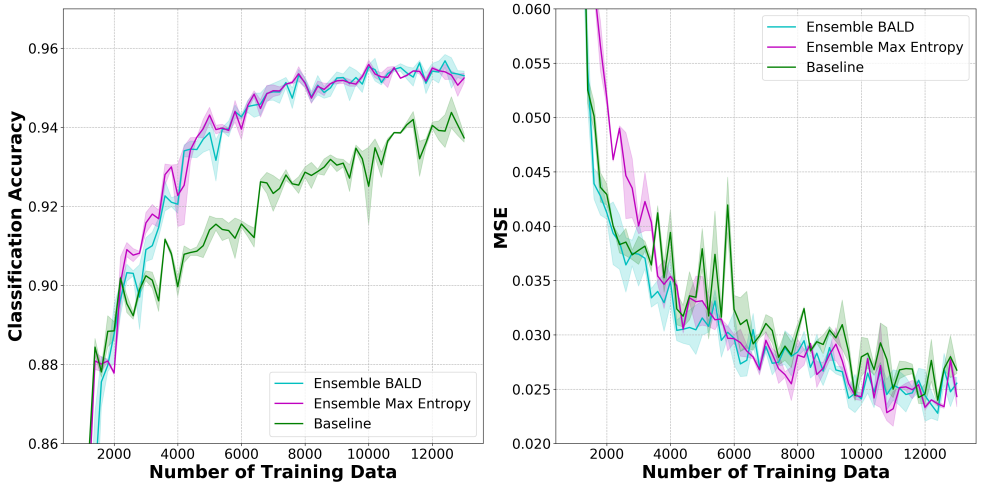
To understand the performance of active learning, we compare the class distribution of queried samples between *Deep Ensembles* + *Max-*

Table 4.1: The number of labeled training data required for the active detector to reach a certain relative error to the full-trained model. The relative classification error is defined as the difference between the classification accuracy of the full-trained model and an active learning model. The relative regression MSE error is defined similarly, but with an extra division by the full-trained model MSE. The table also shows the percentage of labeling efforts saved by the active learning methods compared to the baseline.

	Classification accuracy		Regression MSE	
Relative error to the full-trained model	5%	2%	30%	5%
Baseline	4000	10600	6400	12000
MC Dropout + BALD	3200(20.0%)	4900(53.8%)	**5200(18.8%)**	**9900(17.5%)**
MC Dropout + Maximize Entropy	**2300(42.5%)**	**4200(60.4%)**	5900(7.8%)	10000(16.7%)
Deep Ensembles + BALD	2900(27.5%)	4500(57.5%)	5600(12.5%)	11000(8.3%)
Deep Ensembles + Maximize Entropy	**2300(42.5%)**	4300(59.4%)	5800(9.4%)	10000(16.7%)



(a) *BALD/Maximize Entropy* as acquisition function with *MC Dropout*



(b) *BALD/Maximize Entropy* as acquisition function with *Deep Ensembles*

Figure 4.8: Active learning curves by acquiring data with *BALD* or *Maximize Entropy*. The classification uncertainty estimated by *MC Dropout* and *Deep Ensembles* subsequently

imize Entropy and baseline. This is calculated by taking the difference between the two at each step, normalized by the total number of samples from the corresponding classes in the unlabeled data pool. The results are shown in Fig.4.9. The unlabeled data is highly class-imbalanced as shown in Fig.4.2. However, our method alleviates such problem by actively querying fewer samples from "small vehicle" and more from the other classes. Note that this balancing effect over classes

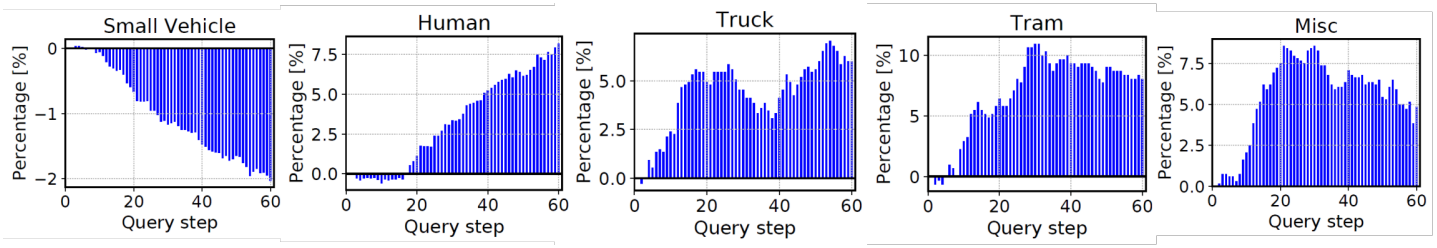


Figure 4.9: Class distribution of queried samples for *Deep Ensembles* + *Maximize Entropy* compared to the baseline method. The vertical axis represents the accumulative queried samples relative to the baseline, normalized by the number of samples of the corresponding classes in the original unlabeled data pool. Compared to the baseline method, active learning queries fewer samples from "small vehicle" and more from the other classes. Note that since there are much more objects in "small vehicle" than the other classes, a small percentage drop in "small vehicle" means a much more percentage rise in other classes.

is due to using uncertainty estimation in queries rather than an ad-hoc solution that explicitly over/under-sample inputs.

4.3 Short Summary

We presented a framework that leverages active learning to efficiently train a multi-modal 3D object detector. The detection model predicts object classification scores and 3D geometric information based on 2D proposals on camera images. We conducted experiments using a "perfect" image detector to compare several query strategies, uncertainty estimation approaches, and acquisition functions. Our active learning method reaches similar detection performance with significantly fewer training samples compared to the baseline method, saving up to 60% labeling efforts. An interesting point is that, in the first few active learning iterations, selecting samples based on only classification uncertainty achieves better results compared to other query criteria, which indicates that classification uncertainty is more expressive for characterizing detection uncertainty given a relatively coarse initial model. By analyzing the class distribution among the query data, we found that querying by classification uncertainty tends to actively choose sub-dominant samples compared to random baseline. Such inclination to balance the training set may help to explain that active learning out-performs passive learning.

Chapter 5

Deep Active Learning for 3D LiDAR Vehicle Detection

By far, we have studied single-task deep active learning with a classifier, and multi-task deep active learning with a refinement network of a deep 3D object detector. The input to both of the active learners contains maximum one object each, which essentially simplifies our design of query strategy, as informativeness estimation and query selection are performed per object. However, it is common for a 3D detector in an autonomous driving system taking in samples containing multiple objects. Therefore, in our last stage experiments, we study the interplay between active learning framework and a complete 3D detector with more complex data and model architecture. To the best of our knowledge, such exploration has never been reported in the literature before.

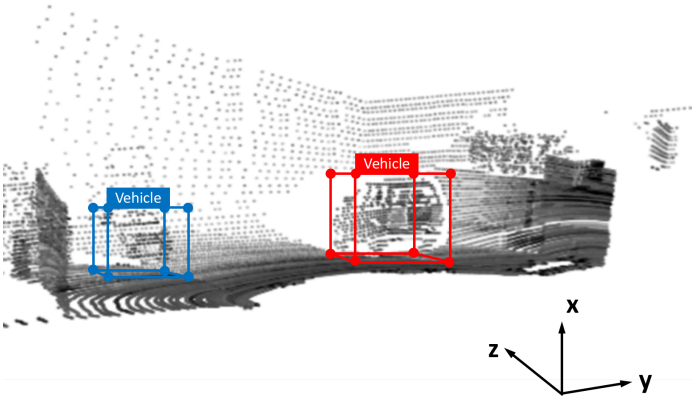
Our active learner feeds on LiDAR point clouds. This is an appealing scenario for active learning since LiDAR data are the hardest and the most expensive to annotate in 3D space. The learner focuses on detecting 3D vehicles which are the most common traffic agents on the road. We introduce the implementation details and experiments in the following sections.

5.1 Implementation Details

5.1.1 Data

In this chapter, we adopt three categories of raw KITTI LiDAR data[20], i.e. city, residential, and road. They cover main urban scenarios for autonomous driving. Each frame of LiDAR point clouds is pre-processed in two steps: first, filtering by elevation, depth, and width, then down-sampling to voxel grids with resolution of 0.1m; second, transforming into BEV feature maps of $M+2$ channels that encode height (M slices), density, and intensity information[10] (Fig.5.1a). The ground truth label for a vehicle object is a 3D oriented bounding box denoted by the 3D coordinates of its 8 corners (Fig.5.1b).

(a) Exemplary BEV LiDAR feature map (left: height map, mid: density map, right: intensity map)



(b) Exemplary 3D bounding box ground truth label

Figure 5.1: Raw LiDAR point clouds are filtered, down-sampled, and transformed into BEV feature maps. Vehicles in each frame of point clouds are labeled with 3D bounding boxes in the LiDAR coordinate system.

We randomly choose 2026 frames of LiDAR point clouds (drive 0001, 0039, 0048, 0070, 0086, 0091) as validation set and 2061 frames (drive 0002, 0015, 0029, 0035, 0057, 0087) as test set. The initial training data size varies under 400 and query step size varies under 100, which are further discussed in the experiment section.

5.1.2 Active 3D Vehicle Detector

We build the active 3D vehicle detector as in [16]. The detector's architecture is shown in Fig.5.2. It basically follows the Faster R-CNN[44] pipeline. To begin with, a backbone network, i.e. ResNet-8[23], extracts high-level features from input LiDAR feature maps (note that we choose this light-weight backbone network mainly to constraint the experiment runtime). The last convolution layer is up-sampled by a factor of two so that small objects can be better detected. An RPN further generates BEV 3D region proposals based on the high-level features. The heights of 3D proposals are of the same value which is selected to be large enough to enclose common vehicles. Three intermediate fully connected layers refine the proposals. Each has 512 hidden neurons and a dropout layer with a dropout rate of 0.5. A regression head predicts final 3D bounding boxes, and a parallel classification head outputs corresponding softmax scores of being "vehicle" or "background".

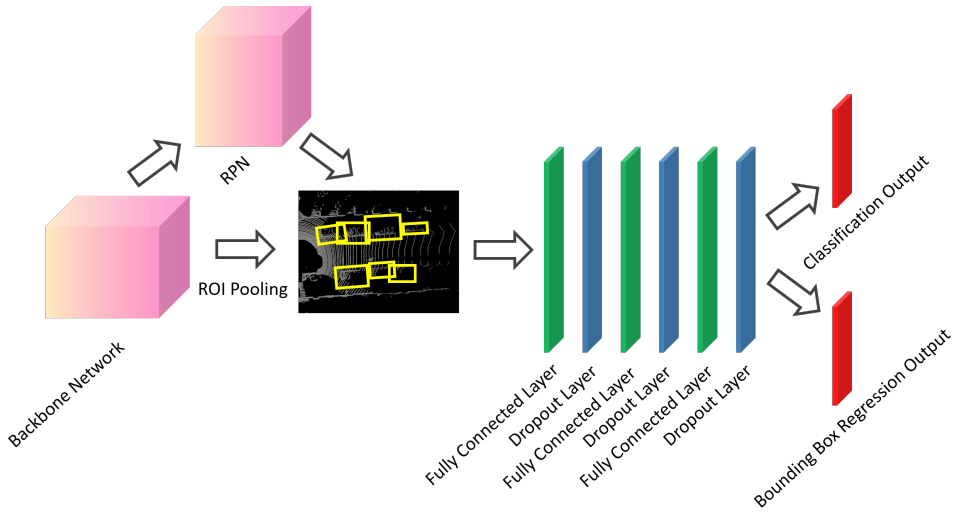


Figure 5.2: Architecture of the 3D LiDAR vehicle detector

The 3D detector is trained with Adam optimizer and a multi-loss function which is the linear sum of classification and regression loss of region proposals and final predictions. The model performance is evaluated by Average Precision at different Intersection over Union (IoU) thresholds.

5.1.3 Learning Loop Formulation

The active learning loop is shown in Fig.5.3. Comparing to the second stage experiments, where the active learner deals with maximally one object per sample and 3D proposals are obtained directly from 2D candidates, our active learner here is an integral 3D detector that handles inputs containing multiple objects in one sample and generates a fixed number of 3D region proposals before final output. In this case, to estimate the sample informativeness, we view each region proposal as a "potential object". As is discussed in the previous chapter, there are three ways to represent the prediction uncertainty on a single potential object: its classification uncertainty, bounding box regression uncertainty, and the linear sum of both. In the following experiments, the two types of output uncertainties are estimated using *MC Dropout* (20 times forward passes) + *BALD* and *Total Variance* (10 times calculation), respectively. In either way, we obtain prediction uncertainty for each region proposal within a single frame and then sum them up as the final prediction uncertainty. Except for summation, there are other options to extract single frame uncertainty from region proposals, such as using the average or maximum. However, we adopt summation as it guarantees the prediction uncertainties are of the same magnitude no matter how many objects lie in the frames. The updated learning loop and algorithm are shown in Figure 5.3 and Algorithm 3 respectively.

5.2 Experiments and Results

In our final stage of experiments, we investigate an active learning framework with the two-stage 3D LiDAR vehicle detector from four aspects. First, we fix the backbone and the RPN network and study the learning behaviors of the refinement network, i.e. the intermediate layers and output heads. The informativeness of each input sample

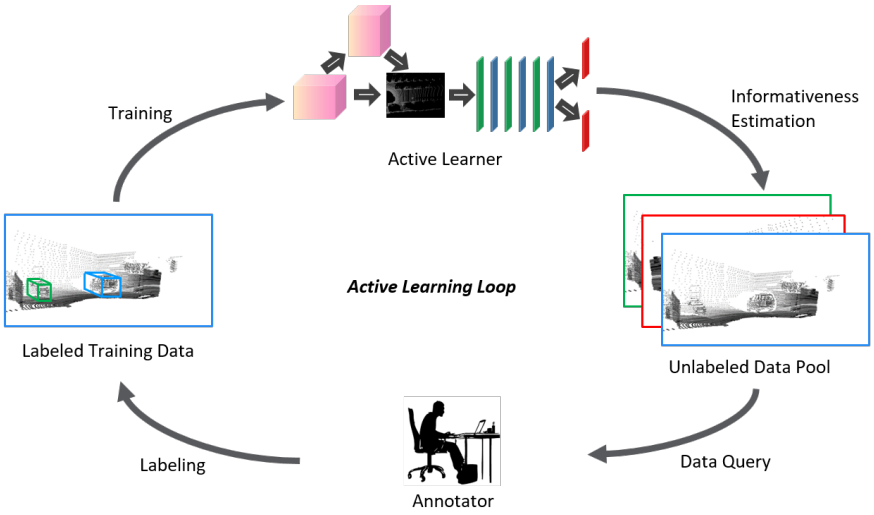


Figure 5.3: Active learning loop for 3D LiDAR detection

is measured by the linear sum of classification (*MC Dropout*) and regression (*TV*) uncertainties of all region proposals. Second, we train the entire network and study its performance. Informativeness is estimated in the same manner. Third, we compare the performances of three query criteria (i.e. query by classification uncertainty, regression uncertainty, and the linear sum of both) with the entire network being trained. Fourth, we analyze the queries for interpretability.

For each experiment, we stop the active learning process after 10 queries as it is highly time-consuming. One single active learning iteration takes 4 - 9 hours on an NVIDIA GeForce GTX 1080 GPU, which makes one complete experiment with 10 queries take 50 - 90 hours. The performance of the network is evaluated by AP at four IoU thresholds, i.e. 0.1, 0.3, 0.5, and 0.7. A full-trained model achieves AP of 0.818, 0.666, 0.359, and 0.036. In the experimental results below, we display the learning curves separately by the thresholds. As usual, x-axis denotes the number of labeled data, and y-axis denotes the performance metric which is AP in this case.

5.2.1 Learning with the Refinement Network Only

The learner is initialized on a training set composed of 70 randomly selected samples. Each active learning iteration queries 20 samples from the unlabeled pool. Fig.5.4 shows that when only training the

Algorithm 3: Deep Active Learning for 3D Vehicle Detection

Input: Initial training dataset X_{train}, unlabeled pool X_{pool}, human/machine annotator A
Result: 3D vehicle detector M
RandomInitialize M
$M = TrainNetwork(X_{train}, M)$
$SaveModelM$ **while** *not StopCondition* **do**
  $U_{cls} = \emptyset$
  $U_{reg} = \emptyset$
  for x **in** X_{pool} **do**
    $ROIs = GenerateROIs(x, M)$
    for ROI **in** $ROIs$ **do**
      ▷ Estimate classification uncertainty
      $u_{cls} = EstClsUncertainty(ROI, M)$
      ▷ Estimate regression uncertainty
      $u_{reg} = EstRegUncertainty(ROI, M)$
      $U_{cls}.append(u_{cls})$
      $U_{reg}.append(u_{reg})$
    end
  end
  $U_{cls} = sum(U_{cls})$ ▷ Sum up uncertainties for all ROI
  $U_{reg} = sum(U_{reg})$
  $X^* = AcquisitionFcn(X_{pool}, U_{cls}, U_{reg})$
  ▷ Query the most informative samples
  $Y^* = Label(X^*, A)$
  $X_{train} = X_{train} \cup \{X^*, Y^*\}$ ▷ Update training dataset
  $X_{pool} = X_{pool} \setminus X^*$ ▷ Update unlabeled pool
  $M = ReTrainNetwork(X_{train}, M)$ ▷ Retrain classifier
  $SaveModel\ M$
end

refinement network with a well-trained backbone and RPN, the performance of both active learning and baseline saturate near the full-

trained model AP with only around 250 labeled samples at IoU threshold of 0.1, 0.3 and 0.5. Active learning by querying the sum of *MC Dropout* classification uncertainty and TV regression uncertainty does not exhibit an advantage over passive learning. This could be due to two reasons. First, the architecture of the learner is too simple. It converges to its maximal performance too fast so that the potential influence of active learning is too weak to appear on the learning curve. For each input sample, the RPN constantly produces 300 region proposals. 250 samples yield 75000 region proposals that easily push the small refinement network to its upper limit. The benefit of active learning, if there is any, could be overwhelmed by the quick saturation. Second, querying by the sum of two types of uncertainties may not be the best fit in this 3D LiDAR detection scenario. As is mentioned in [49], the active learning settings could vary case by case. More query criteria shall be explored. We continue the discussion in the following sections.

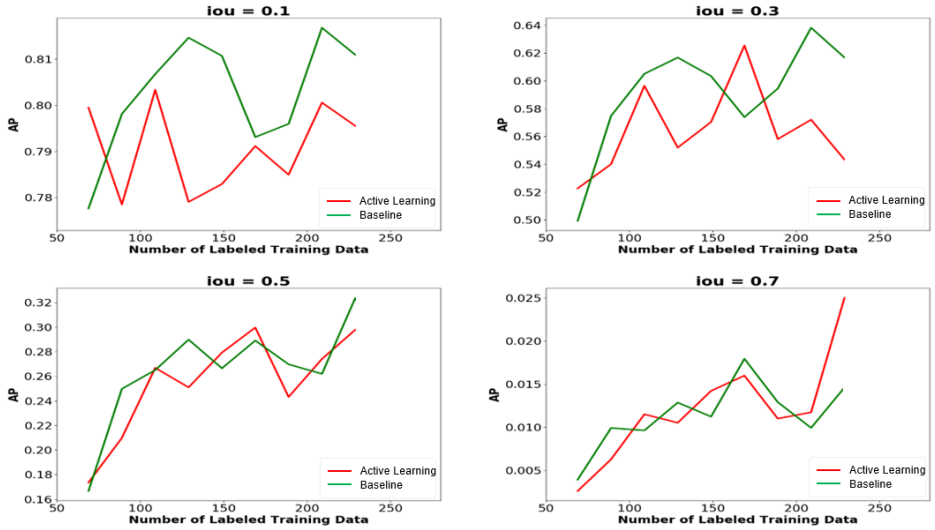


Figure 5.4: Active learning curves with refinement network (intermediate fully connected layers + output heads). The backbone and RPN are well-trained and fixed during the active loops. Query data are selected based on the weighted sum of MC BALD classification uncertainty and bounding box regression uncertainty.

5.2.2 Learning with the Entire 3D Detection Network

By involving backbone and RPN in the learning loop, the complexity of the learner is increased. Consequently, we raise the initial training data size to 225, and query size to 100. The learning curves in Fig.5.5 indicates a performance drop given the same quantity of labeled data. Naturally, more training data is needed to saturate the detection performance. Nevertheless, active learning still fails to show significant improvements - only a slight margin over baseline when the amount of labeled data is below 800.

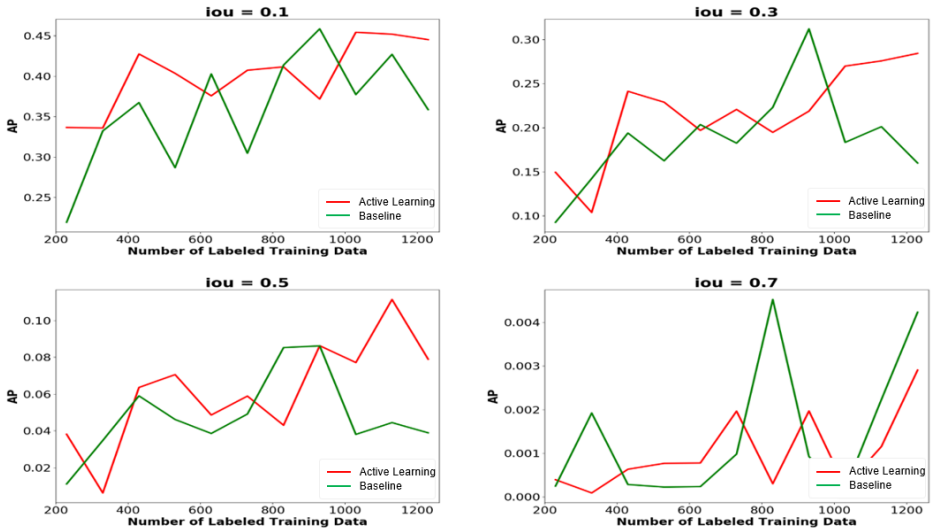


Figure 5.5: Active learning curves with the entire network. Query data are also selected based on the weighted sum of MC BALD classification uncertainty and bounding box regression uncertainty.

5.2.3 Query Criteria

We compare three query criteria in Fig.5.6. Despite the large variance, we can see there is still no significant difference between switching query strategy.

There are two possible reasons that active learning does not work as expected. First, considering that the active learner out-performs baseline given relatively high quality ROIs in Chapter 4, our backbone (ResNet-8) here is too weak to provide reliable ROIs, so that the prediction uncertainties estimated based on ROIs are too noisy to charac-

terize the true uncertainty. Second, even with relatively accurate ROIs, the summation of all ROI uncertainties may not be the best way to represent prediction uncertainty for our two-stage 3D detection model. Intuitively, it will query samples with less ground truth objects in a single frame, as false positive ROIs are likely to give higher uncertainty values. However, Brust et al.[6] provides an alternative with opposite intuition that samples with more ground truth objects shall be queried. They study active learning with a one-stage 2D detector, and show that summing up the uncertainties of the final outputs for query yields better results than random sampling. Therefore, despite that several uncertainty estimation methods have been proved to be effective for single objects in the previous stages of experiment, we need to develop better ways of aggregating uncertainties of multiple objects in a single frame. We leave this as an interesting future research topic.

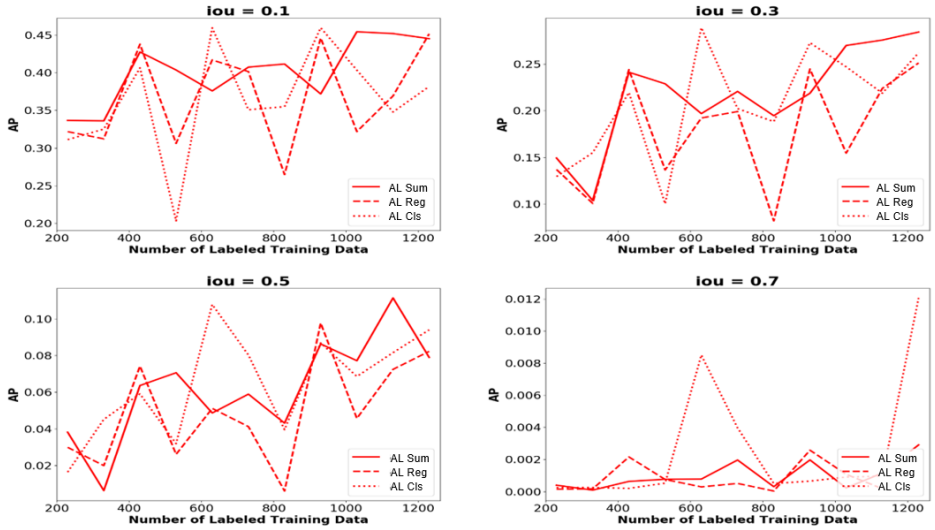


Figure 5.6: Active learning curves with three query criteria: query by classification uncertainty, regression uncertainty, and weighted sum of both, denoted by *AL Cls*, *AL Reg*, and *AL Sum*, respectively.

5.2.4 Query Analysis

We first show the underlying feature distribution of the object vehicles from our dataset in Fig.5.7. In all three scenarios (road, residential, and

city), the features are significantly biased. The dominant distance lies in 10 - 30 meters, the dominant orientation is around $0°$ and ± 180, and the dominant occlusion status is being visible to the LiDAR.

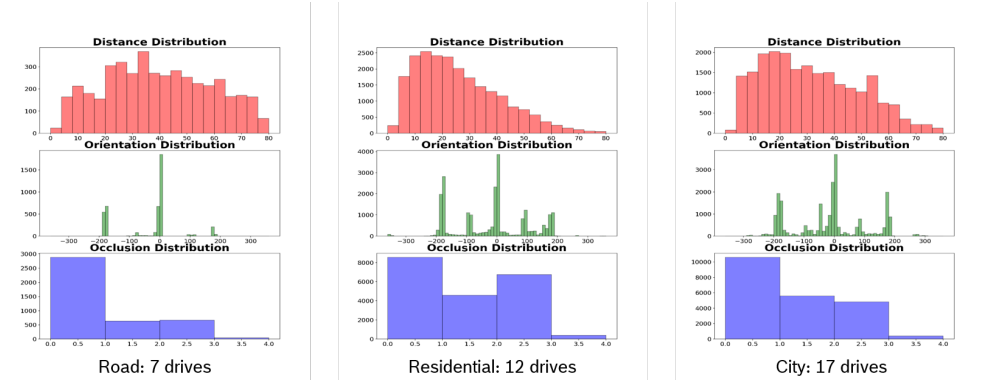


Figure 5.7: Underlying feature distribution of object vehicles. Based on KITTI raw data, our entire dataset is composed of 7 drives of "road" data, 12 drives of "residential" data, and 17 drives of "city" data. The feature distributions of object vehicles from these three scenarios are displayed separately. In each column, we show the distribution of distance, orientation, and occlusion status relative to the ego vehicle. X-axes denote meters, degrees, or visibility (better visibility with smaller numbers). Y-axes denotes the number of frames.

Fig.5.8 displays the query data from the first and second active learning and baseline iteration. The entire detector is trained and queries are selected based on the uncertainty sum. For visual convenience, we use the RGB images of the same frames instead of the input LiDAR samples. The queried instances by the active learner are arranged in informativeness descent order, i.e. left-upper corner is the most informative, while right-bottom corner is the least. By comparing active and passive queries, we can tell that active learning tends to select samples with less dominant features, e.g. samples with objects that are not parallel to the ego vehicle. After the first learning iteration (Fig.5.8a), the active learner queries 7 frames that contain non-parallel vehicles among the most informative 20 samples, while the passive learner only queries 4. After the second iteration (Fig.5.8b), the numbers become 11 to 3. Similar biased queries are also observed in the iterations afterward and other active learning experiments.

(a) Baseline (up) and active (bottom) queries from the first learning iteration.

(b) Baseline (up) and active (bottom) queries from the second learning iteration.

Figure 5.8: Exemplary queries from an active learning experiment. The entire 3D vehicle is trained in the learning loop. The queries are selected based on the weighted sum of both classification and regression uncertainties. Active queries are arrayed in informativeness descent order. Frames that contain object vehicles non-parallel to the ego-vehicle are highlighted with red rectangles.

5.3 Short Summary

In the last stage, we explore deep active learning with an integral 3D LiDAR vehicle detector. We empirically show that the active detector also displays a tendency of querying samples with sub-dominant samples. However, the most effective uncertainty estimation method, acquisition function, and query criteria in previous stages of experiments fail to make distinguishable improvement over the baseline method. There could be two reasons. First, the ROIs produced by a weak backbone is not far from random predictions, so that the estimated uncertainties based on the ROIs are noisy and nearly meaningless. Second, the aggregation of object uncertainties within a single frame shall be more carefully designed. Our experiments also reveals another challenge of deep active learning for 3D detection that the experimental investigation requires a huge amount of time and computation resources. With limited GPUs and time, we can only test with ResNet-8 backbone, a few times of active learning iterations, and several query settings. Without doubt, heavy resources are needed for a more comprehensive research on deep active learning for 3D detection in the future.

Chapter 6

Conclusions

This thesis is motivated by the data labeling challenge in the autonomous driving industry. To achieve great environment perception, the 3D object detection model on an autonomous vehicle needs to be constantly updated with enormous amount of labeled data. However, the data are highly expensive and tedious to label, especially 3D LiDAR point clouds. In order to alleviate the 3D annotation cost, we develop and study deep active learning frameworks that enable the neural network models to estimate the informativeness of unlabeled data by means of uncertainty and only select the most informative samples for labeling. Our work is pioneering in the sense of combining active learning with a 3D deep neural network object detector. Without bells and whistles, we proposed a deep active learning framework in our second stage of experiments that is able to train a multi-modal 3D detector with up to 60% less annotated data compared to the random sampling baseline method. This is attributed to high-quality region proposals synthesized for detection and effective estimation of model prediction uncertainties.

This thesis is composed of three progressive stages of study. They are designed based on two facts. First, a 3D detection task can be essentially decomposed into two sub-tasks: object classification and 3D bounding box regression. Second, a state-of-the-art 3D detector can be built following a two-stage pipeline. On such basis, we start the first stage experiments with single-task active learning for classification. The results indicate the importance of a well-initialized model. In the second stage, we move on to multi-task active learning. A refinement network of a two-stage multi-modal 3D object detector is updated in

the active learning loop that significantly reduces annotation cost for training. The network is fed with 3D region proposals generated from high-quality 2D image proposals, and the entropy-based classification uncertainty effectively capture the informativeness of the inputs. In the third and also the final stage, we incorporate an integral two-stage 3D vehicle detection network in the active learning loop. Query data are selected based on the aggregated uncertainties of ROIs. However, active learning does not show recognizable improvements over baseline in our experiments probably due to the lack of a capable backbone and a more appropriate query strategy.

From our experiments, we have concluded four takeaways for future work. First, the initial model matters. A good initial model improves the quality of uncertainty estimation, alleviates bias, and helps the active learner out-run the random sampling baseline from the beginning. Second, entropy based classification uncertainty is more useful than variance based regression uncertainty for a neural network detector to actively select queries. However, neither different entropy based acquisition functions (*Max Entropy* and *BALD*) nor different classification uncertainty estimation approaches (*MC Dropout* and *Ensembles*) show significant differences in performance. Third, high quality ROIs are crucial for DAL3D. From the second to the third stage, the "perfect" image detector is substituted with a relatively weak backbone model, which downgrades the quality of ROIs and uncertainty estimation. We speculate this is the main reason that leads to the performance drop of active learning in the third stage. Last but not least, we find that an active learner tends to increase the diversity of training data by querying samples with sub-dominant features. This is achieved automatically based on evaluating the informativeness of inputs rather than predetermined rules, which offers an interesting perspective to understand why active learning works.

Finally, we would like to propose several directions for future research. From a practical perspective, more computational resource and more efficient implementation of active learning algorithms are necessary for conducting comprehensive study in this field. Training a deep neural network learner with many active learning iterations already take significant amount of time, let alone one has to experiment repetitively with versatile active learning settings. Also, in this thesis, we work with a limited number of data in the unlabeled pool. However, active learning is also valuable in "life-long learning" appli-

cations where we want to update the learner on the air (for instance, to continuously improve an autonomous driving system based on user data collected online), which requires very different system pipeline and involves multi-disciplinary researches on incremental or continual learning. From a more theoretical perspective, there still lacks more rigorous guidelines for configuring active learning frameworks. For example, one usually needs to do a lot of trial-and-error to find the best-fit configurations. We mainly study and compare multiple uncertainty sampling strategies for training an active 3D detector in this thesis, yet many other strategies await for exploration, such as query-by-committee or geometric methods, just to name a few. Also, in the light of certain complex learning tasks (3D detection for instance), one may face the problem of aggregating the estimated informativeness before making queries. Very scarce literature has touched this topic by far. Last but not least, it is noteworthy to mention that, similar to other machine learning technologies, active learning might lead to job losses. Less data are needed to be labeled, which is likely to cause unemployment of human annotators. It is commonly acknowledged that the balance between machine intelligence and human labor is a social challenge we have to face in the following decades. We believe that researcher, engineers, and other social parties shall keep such ethical concerns in mind along with the pursuit of technology advances. In all, we look forward to more inspiring and exciting work on active learning for 3D detection and beyond that coincides with social benefits in general.

Bibliography

[1] National Highway Traffic Safety Administration. *Automated Vehicles for Safety*. https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety. [Online; accessed 05-Dec-2018]. 2018.

[2] Alireza Asvadi et al. "Multimodal vehicle detection: fusing 3d-lidar and color camera data". In: *Pattern Recognition Letters* 115 (2018), pp. 20–29.

[3] Jorge Beltrán et al. "BirdNet: A 3D object detection framework from LiDAR information". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 3517–3523.

[4] William H Beluch et al. "The power of ensembles for active learning in image classification". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 9368–9377.

[5] Klaus Bengler et al. "Three decades of driver assistance systems: Review and future perspectives". In: *IEEE Intelligent Transportation Systems Magazine* 6.4 (2014), pp. 6–22.

[6] Clemens-Alexander Brust, Christoph Käding, and Joachim Denzler. "Active learning for deep object detection". In: *arXiv preprint arXiv:1809.09875* (2018).

[7] Luca Caltagirone et al. "Fast LIDAR-based road detection using fully convolutional neural networks". In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2017, pp. 1019–1024.

[8] Chenyi Chen et al. "Deepdriving: Learning affordance for direct perception in autonomous driving". In: *Computer Vision (ICCV), 2015 IEEE International Conference on*. IEEE. 2015, pp. 2722–2730.

[9] Xiaozhi Chen et al. “3d object proposals for accurate object class detection”. In: *Advances in Neural Information Processing Systems*. 2015, pp. 424–432.

[10] Xiaozhi Chen et al. “Multi-view 3d object detection network for autonomous driving”. In: *IEEE CVPR*. Vol. 1. 2. 2017, p. 3.

[11] Feras Dayoub, Niko Sunderhauf, and Peter I Corke. “Episode-based active learning with Bayesian neural networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2017, pp. 26–28.

[12] Xinxin Du, Marcelo H Ang, and Daniela Rus. “Car detection for autonomous vehicle: LIDAR and vision fusion approach through deep learning framework”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 749–754.

[13] Xinxin Du et al. “A general pipeline for 3d detection of vehicles”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 3194–3200.

[14] Martin Engelcke et al. “Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 1355–1361.

[15] Seyda Ertekin et al. “Learning on the border: active learning in imbalanced data classification”. In: *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM. 2007, pp. 127–136.

[16] Di Feng, Lars Rosenbaum, and Klaus Dietmayer. “Towards Safe Autonomous Driving: Capture Uncertainty in the Deep Neural Network For Lidar 3D Vehicle Detection”. In: *arXiv preprint arXiv:1804.05132* (2018).

[17] Yarin Gal. “Uncertainty in deep learning”. In: *PhD thesis, University of Cambridge* (2016).

[18] Yarin Gal and Zoubin Ghahramani. “Dropout as a bayesian approximation: Representing model uncertainty in deep learning”. In: *international conference on machine learning*. 2016, pp. 1050–1059.

[19] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. "Deep bayesian active learning with image data". In: *arXiv preprint arXiv:1703.02910* (2017).

[20] Andreas Geiger et al. "Vision meets robotics: The KITTI dataset". In: *The International Journal of Robotics Research* 32.11 (2013), pp. 1231–1237.

[21] Ross Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587.

[22] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[23] Kaiming He et al. "Identity mappings in deep residual networks". In: *European conference on computer vision*. Springer. 2016, pp. 630–645.

[24] Stephan Heinrich. "Flash Memory in the emerging age of autonomy". In: *Proc. Flash Memory Summit* (2017).

[25] Prateek Jain and Ashish Kapoor. "Active learning for large multi-class problems". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, pp. 762–769.

[26] Chieh-Chi Kao et al. "Localization-aware active learning for object detection". In: *arXiv preprint arXiv:1801.05124* (2018).

[27] Ashish Kapoor et al. "Gaussian processes for object categorization". In: *International journal of computer vision* 88.2 (2010), pp. 169–188.

[28] *Keras: Deep Learning for humans*. `https://github.com/keras-team/keras`.

[29] Jason Ku et al. "Joint 3d proposal generation and object detection from view aggregation". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1–8.

[30] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. "Simple and scalable predictive uncertainty estimation using deep ensembles". In: *Advances in Neural Information Processing Systems*. 2017, pp. 6402–6413.

[31] Yann LeCun. "The MNIST database of handwritten digits". In: *http://yann. lecun. com/exdb/mnist/* (1998).

[32] David D Lewis and William A Gale. "A sequential algorithm for training text classifiers". In: *SIGIR'94*. Springer. 1994, pp. 3–12.

[33] Bo Li. "3D fully convolutional network for vehicle detection in point cloud". In: *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE. 2017, pp. 1513–1518.

[34] Bo Li, Tianlei Zhang, and Tian Xia. "Vehicle detection from 3d lidar using fully convolutional network". In: *arXiv preprint arXiv:1608.07916* (2016).

[35] Michael Lindenbaum, Shaul Markovitch, and Dmitry Rusakov. "Selective sampling for nearest neighbor classifiers". In: *Machine learning* 54.2 (2004), pp. 125–152.

[36] Wei Liu et al. "Ssd: Single shot multibox detector". In: *European conference on computer vision*. Springer. 2016, pp. 21–37.

[37] Damien Matti, Hazım Kemal Ekenel, and Jean-Philippe Thiran. "Combining lidar space clustering and convolutional neural networks for pedestrian detection". In: *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE. 2017, pp. 1–6.

[38] Kazuki Minemura et al. "LMNet: Real-time Multiclass Object Detection on CPU Using 3D LiDAR". In: *2018 3rd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*. IEEE. 2018, pp. 28–34.

[39] Charles R Qi et al. "Frustum pointnets for 3d object detection from rgb-d data". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 918–927.

[40] Charles R Qi et al. "Pointnet: Deep learning on point sets for 3d classification and segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 652–660.

[41] Guo-Jun Qi et al. "Two-dimensional active learning for image classification". In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2008, pp. 1–8.

[42] Joseph Redmon and Ali Farhadi. "Yolov3: An incremental improvement". In: *arXiv preprint arXiv:1804.02767* (2018).

[43] Joseph Redmon et al. "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[44] Shaoqing Ren et al. "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[45] Giuseppe Riccardi and Dilek Hakkani-Tur. "Active learning: Theory and applications to automatic speech recognition". In: *IEEE transactions on speech and audio processing* 13.4 (2005), pp. 504–511.

[46] Nicholas Roy and Andrew McCallum. "Toward optimal active learning through monte carlo estimation of error reduction". In: *ICML, Williamstown* (2001), pp. 441–448.

[47] Soumya Roy, Vinay P Namboodiri, and Arijit Biswas. "Active learning with version spaces for object detection". In: *arXiv preprint arXiv:1611.07285* (2016).

[48] Soumya Roy, Asim Unmesh, and Vinay P Namboodiri. "Deep active learning for object detection". In: *British Machine Vision Conference*. 2018, pp. 3–6.

[49] Burr Settles. *Active learning literature survey*. Tech. rep. University of Wisconsin-Madison Department of Computer Sciences, 2009.

[50] H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. "Query by committee". In: *Proceedings of the fifth annual workshop on Computational learning theory*. ACM. 1992, pp. 287–294.

[51] Sayanan Sivaraman and Mohan M Trivedi. "Active learning for on-road vehicle detection: A comparative study". In: *Machine vision and applications* 25.3 (2014), pp. 599–611.

[52] Sayanan Sivaraman and Mohan Manubhai Trivedi. "A general active-learning framework for on-road vehicle recognition and tracking". In: *IEEE Transactions on Intelligent Transportation Systems* 11.2 (2010), pp. 267–276.

[53] Petru Soviany and Radu Tudor Ionescu. "Frustratingly Easy Trade-off Optimization between Single-Stage and Two-Stage Deep Object Detectors". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 0–0.

[54] Sudheendra Vijayanarasimhan and Kristen Grauman. "Large-scale live active learning: Training object detectors with crawled data and crowds". In: *International Journal of Computer Vision* 108.1-2 (2014), pp. 97–114.

[55] Carl Vondrick and Deva Ramanan. "Video annotation and tracking with active learning". In: *Advances in Neural Information Processing Systems*. 2011, pp. 28–36.

[56] Dominic Zeng Wang and Ingmar Posner. "Voting for Voting in Online Point Cloud Object Detection." In: *Robotics: Science and Systems*. Vol. 1. 3. 2015, pp. 10–15607.

[57] Meng Wang and Xian-Sheng Hua. "Active learning in multimedia annotation and retrieval: A survey". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 2.2 (2011), p. 10.

[58] Philip D Wasserman and Tom Schwartz. "Neural networks. II. What are they and why is everybody so interested in them now?" In: *IEEE Expert* 3.1 (1988), pp. 10–15.

[59] Han Xiao, Kashif Rasul, and Roland Vollgraf. "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms". In: *arXiv preprint arXiv:1708.07747* (2017).

[60] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. "Pointfusion: Deep sensor fusion for 3d bounding box estimation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 244–253.

[61] Songbai Yan, Kamalika Chaudhuri, and Tara Javidi. "Active Learning with Logged Data". In: *arXiv preprint arXiv:1802.09069* (2018).

[62] Yi Yang et al. "Multi-class active learning by uncertainty sampling with diversity maximization". In: *International Journal of Computer Vision* 113.2 (2015), pp. 113–127.

[63] Angela Yao et al. "Interactive object detection". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2012, pp. 3242–3249.

[64] Yin Zhou and Oncel Tuzel. "Voxelnet: End-to-end learning for point cloud based 3d object detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4490–4499.

TRITA -EECS-EX-2019:798