Deep learning multi-modal fusion based 3D object detection

Haodi Zhang

Normandie Université

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN NORMANDIE

THÈSE

Pour obtenir le diplôme de doctorat

Spécialité INFORMATIQUE

Préparée au sein de l'INSA Rouen Normandie

Deep learning multi-modal fusion based 3D object detection

Présentée et soutenue par
HAODI ZHANG

Thèse soutenue le 01/06/2023 devant le jury composé de		
MME SYLVIE CHAMBON	MAITRE DE CONFERENCES DES UNIVERSITES HDR, INP Toulouse	Rapporteur
M. DOMINIQUE GRUYER	DIRECTEUR DE RECHERCHE, University Gustave Eiffel	Rapporteur
MME FARAH CHEHADE	MAITRE DE CONFERENCES DES UNIVERSITES HDR, Université de Technologie de Troyes	Membre
M. PAUL HONEINE	PROFESSEUR DES UNIVERSITÉS, Université de Rouen Normandie	Membre
MME ALEXANDRINA ROGOZAN	MAÎTRE DE CONFÉRENCES, Institut national des sciences appliquées de Rouen	Membre Co-encadrant
M. FAWZI NASHASHIBI	DIRECTEUR DE RECHERCHE, INRIA	Président du jury
M. ABDELAZIZ BENSRHAIR	PROFESSEUR DES UNIVERSITÉS, Institut national des sciences appliquées de Rouen	Directeur de thèse

Thèse dirigée par ABDELAZIZ BENSRHAIR (Laboratoire d'Informatique, du Traitement de l'Information et des Systèmes)

MI/IΣ

litis

Acknowledgements

I would like to express my sincere gratitude to my two supervisors Prof. Dr. Abdelaziz Bensrhair and Assoc. Prof. Dr. Alexandrina Rogozan. Prof. Dr. Abdelaziz Bensrhair gives me full trust and always encourages me to explore positively. He provides me with a lot of help through all my PhD thesis. Assoc. Prof. Dr. Alexandrina Rogozan is an outstanding female scientist who helps me finding the key to solving the problems under complex conditions. It is an honor to work with both of you.

Thanks to China Scholarship Council for providing me with financial support. This provides me an invaluable opportunity to interact in depth with international research scientists. Chinese Embassy in France is also concerned about our studies and health. During the epidemic, they prepare supplies for our students. The French government and INSA Rouen Normandie provide a very friendly study environment for our international students.

LITIS Lab is a very inclusive family for me. We have a diverse group of researchers who have made outstanding contributions in a variety of fields. The lab often organizes different seminars to bring together the wisdom of the group and to spark new thoughts. The staff of our lab are friendly and helpful. I would like to give special thanks to Jean-François Brulard, Brigitte Diarra and Sandra Hague. They have been indispensable in helping me overcome the cultural differences.

I want to say thank you to my PhD thesis reviewers and my jury. Your comments on my thesis have made my research even more complete. Your wealth of expertise will broaden my mind.

I have many friends who have helped me in the exploration of my PhD thesis. They are Yan Yujin, Zhang Jing, Jia Linlin, Wang Xuefei, Wang Jundong, Xu jie, Ren Chao, Shi Lujie, Dănuţ Ovidiu Pop, Robin Condat, Nicolas Brulard and Marie-Véronique BRULARD.

Finally, I would like to emphasize my gratitude to my family. It was their selfless

support that made it possible for me to be where I am today. My work is a gift to my family in China. I hope to make my contribution to my family and society in the near future. Thanks to all.

Résumé

La détection d'objets en 3D est un élément clé du module de perception du véhicule autonome. Après la détection, la position spatiale de l'objet est indiquée dans une boîte de délimitation cubique. Les tâches ultérieures du pipeline, telles que la reconnaissance, la segmentation et la prédiction, reposent sur une détection précise. Au cours des cinq dernières années, la détection d'objets en 3D a suscité de plus en plus d'attention. De nombreux algorithmes excellents basés sur l'apprentissage profond ont été proposés et ont permis de réaliser des progrès significatifs en matière de précision de détection.

Diverses modalités de données sont disponibles pour la détection d'objets en 3D, l'image et le LiDAR étant les deux modalités les plus couramment adoptées. La modalité LiDAR est préférée par la plupart des détecteurs en raison de la précision de ses informations de profondeur qui délimitent spatialement l'objet. En revanche, la modalité image est limitée par l'ambiguïté des informations de profondeur, d'où une précision insuffisante de la détection d'objets en 3D. Certains travaux pionniers tentent d'exploiter à la fois les informations de profondeur précises et les riches informations sémantiques en fusionnant les deux modalités. Cependant, il n'existe pas encore de paradigme de fusion dont l'efficacité a été largement prouvée. En outre, toutes les méthodes existantes de détection d'objets 3D par fusion reposent sur l'hypothèse de données synchronisées. Plus le nombre de modalités augmente, plus la fréquence de synchronisation diminue, ce qui entraîne un goulot d'étranglement dans l'efficacité de la détection. Cela réduira évidemment la sécurité des véhicules autonomes.

Pour répondre aux questions ci-dessus, cette thèse propose les contributions suivantes, résumées en quatre points : 1) La distorsion lors de l'évaluation de la note de précision moyenne en utilisant la méthode d'interpolation à N points est révélée. La distorsion de la précision moyenne qui peut conduire à l'échec de l'évaluation du serveur est complètement analysée. Afin de résoudre le problème de distorsion, nous proposons une méthode d'interpolation à N points améliorée. En modifiant la méthode de calcul de la zone de l'intervalle d'interpolation et l'emplacement du point d'interpolation, la distorsion de la précision moyenne est correctement éliminée. 2) Nous introduisons un modèle d'optimisation d'image unifié pour supprimer les zones redondantes qui partagent les mêmes images de paramètres échantillonnés. Au lieu d'optimiser chaque image, l'algorithme NPAE proposé estime et récolte la zone non piétonne commune pour toutes les images. Par conséquent, l'algorithme NPAE peut réduire la consommation de temps de détection tout en maintenant la précision de détection. 3) Ensuite, nous explorons la méthode de fusion tardive des données multimodales synchrones. Un grand nombre de faux positifs sont observés dans les propositions de détection d'objets 3D basées sur la seule modalité LiDAR. Ces propositions de faux positifs peuvent être classées dans la modalité image. Par

conséquent, toutes les propositions sont projetées sur le plan de l'image pour vérification par le classificateur d'image. Ce modèle de fusion multi-modale tardive est appelé vérification cross-modale (CMV). Après le traitement du modèle CMV, les faux positifs sont réduits de 50 %. 4) Enfin, nous découvrons un nouveau scénario de fusion multimodale asynchrone et proposons une solution. Le flux de données asynchrone est largement présent dans les procédures d'échantillonnage de capteurs pour des modalités multiples. L'utilisation complète des données asynchrones permet d'augmenter de manière significative la fréquence des données fournies pour la détection par un véhicule autonome. Dans ce but, nous proposons le détecteur d'objets 3D à fusion multimodale asynchrone (AF3D). Il peut fonctionner à la fois en état synchrone et asynchrone. De plus, AF3D a la capacité de fusionner des données asynchrones avec des données synchrones. Dans ce cas, la précision de détection de l'image asynchrone avec seulement la modalité image est considérablement améliorée.
Mots-clés: Détection d'objets en 3D, fusion multimodale, fusion asynchrone, précision moyenne, apprentissage profond, véhicule autonome.

Abstract

3D object detection is a key component of the autonomous vehicle perception module. The spatial position of the object is indicated in a cubic bounding box after detection. The subsequent pipeline tasks, such as recognition, segmentation and prediction, rely on an accurate detection. Over the past five years, 3D object detection has been gaining more and more attention. Many excellent deep learning-based algorithms have been proposed and have led to significant progress in detection accuracy.

Various data modalities are available for 3D object detection, where image and LiDAR are the two commonly adopted modalities. LiDAR modality is preferred by most detectors since its accurate depth information that spatially contours the object. In contrast, image modality is limited by the ambiguous depth information, resulting in unsatisfactory 3D object detection accuracy. Some pioneering work attempts to exploit both accurate depth information and rich semantic information by fusing the two modalities. However, there is not yet a fusion paradigm that has been widely proven to be effective. In addition, all existing fusion 3D object detection methods are based on the assumption of synchronized data. As the number of modalities increases, the synchronization frequency decreases leading to a bottleneck in the detection efficiency. It will obviously reduce the safety of autonomous vehicles.

To address the above issues, this thesis proposes the following contributions, summarized in four points : 1) The distortion during the evaluation of average precision score using the N-point interpolation method is revealed. The average precision distortion that may lead to server evaluation failure is completely analyzed. In order to address the distortion problem, we propose an enhanced N-point interpolation method. By changing the interpolation interval area calculation method and interpolation point location, the average precision distortion is properly eliminated. 2) We introduce a unified image optimization model for removing redundant areas that share the same sampled parameter images. Instead of optimizing each image, the proposed NPAE algorithm estimates and crops the common Non-Pedestrian Area for all images. Therefore, the NPAE algorithm can reduce the detection time consumption while maintaining detection accuracy. 3) Afterwards, We explore the late fusion method of synchronous multi-modal data. A large number of false positives are observed in the 3D object detection proposals based on the single LiDAR modality. These false positive proposals can be classified in image modality. Therefore, all proposals are projected onto image plane for verification by the image classifier. This late multi-modal fusion model is called Cross-Modal Verification (CMV). After the processing of CMV model, the false positives are decreased by 50%. 4) Finally, we discover a novel asynchronous multi-modal fusion scenario and provide a solution. Asynchronous data flow is widely present in sensor sampling procedures for multiple modalities. Making full utilization of asynchronous data en-

ables a significant increase in the frequency of data supplied for detection by autonomous vehicle. For this purpose, we propose the Asynchronous Multi-modal Fusion 3D object detector (AF3D). It can work in both synchronous and asynchronous states. Furthermore, AF3D has the ability to fuse asynchronous data with synchronous data. In this case, the detection accuracy of asynchronous frame with only image modality is significantly improved.
Keywords: 3D object detection, Multi-modal fusion, Asynchronous fusion, Average precision, Deep learning, Autonomous vehicle

List of Publications

During this Ph.D., the following research works have been published:

Peer-reviewed journal papers

H. Zhang*, A. Rogozan and A. Bensrhair. "An Enhanced N-Point Interpolation Method to Eliminate Average Precision Distortion". ***Pattern Recognition Letters***. vol. 158, pp. 111-116, 2022, doi: 10.1016/j.patrec.2022.04.028, JCR Impact factor : 3.756.

H. Zhang*, D. O. Pop, A. Rogozan and A. Bensrhair, "Accelerate High Resolution Image Pedestrian Detection With Non-Pedestrian Area Estimation". ***IEEE Access***, vol. 9, pp. 8625-8636, 2021, doi: 10.1109/ACCESS.2021.3049401, JCR Impact factor : 3.367.

Peer-reviewed international conference papers

H. Zhang*, A. Rogozan and A. Bensrhair. "Cross-modal verification for 3D object detection". In conference: ***European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*** (ESANN), Bruges (Belgium), 6-8 October 2021, doi: 10.14428/esann/2021.ES2021-97.

Peer-reviewed national conference papers

H. Zhang*, A. Rogozan and A. Bensrhair. "Acceleration of Pedestrian Detection in High Resolution Image". In ***Groupe de Recherche et d'Etudes de Traitement du Signal et des Images*** (GRETSI), Nancy (French), 6-9 Septembre 2022.

Papers prepared for peer-reviewed journals

H. Zhang*, A. Rogozan and A. Bensrhair. AF3D: Asynchronous multi-modal fusion for 3D object detection. Pattern Recognition Letters. Submitted and under review.

List of Abbreviations

Chapter 1

AV	**A**utonomous **V**ehicle
SAE	**S**ociety of **A**utomotive **E**ngineers
LiDAR	**Li**ght **D**etection **A**nd **R**anging
IMU	**I**nertial **M**easurement **U**nit
GPS	**G**lobal **P**ositioning **S**ystem
2/3 D	**2/3 D**imensional
CNN	**C**onvolutional **N**eural **N**etworks
AP	**A**verage Precision
NPAE	**N**on-**P**edestrian **A**rea **E**stimation
CMV	**C**ross-**M**odal **V**erification
AF3D	**A**synchronous multi-modal **F**usion for **3D** object detection

Chapter 2

MCP	**M**c**C**ulloch-**P**itts **M**odel
GPU	**G**raphics **P**rocessing **U**nit
BN	**B**atch **N**ormalization
NLP	**N**atural **L**anguage **P**rocessing
MLP	**M**ultiple **L**ayer **P**erceptron
ReLU	**Re**ctified **L**inear **U**nit
FC	**F**ully **C**onnected layer
SGD	**S**tochastic **G**radient **D**escent
SIFT	**S**cale-**I**nvariant **F**eature **T**ransform
HOG	**H**istogram of **O**riented **G**radients
RCNN	**R**egions with **CNN** features
mAP	**m**ean **A**verage **P**recision

SVM	**S**upport **V**ector **M**achine
NMS	**N**on-**M**aximum **S**uppression
SPM	**S**patial **P**yramid **M**atching
SPM	**S**patial **P**yramid **M**atching
RoI	**R**egion **o**f **I**nteresting
RPN	**R**egion **P**roposal **N**etwork
fps	**f**rame **p**er **s**econd
FPN	**F**eature **P**yramid **N**etwork
FPS	**F**arthest **P**oint **S**ampling
SA	**S**et **A**bstract
VFE	**V**oxel **F**eature **E**ncoding
VSA	**V**oxel **S**et **A**bstraction
PR	**P**recision **R**ecall
IoU	**I**ntersection **o**ver **U**nion
BBox	**B**ounding **Box**
AP	**A**verage **P**recision
TP	**T**rue **P**ositive
FP	**F**alse **P**ositive
FN	**F**alse **N**egative

Chapter 3

API	**A**verage **P**recision **I**ncrease curve
BEV	**B**ird **E**ye's **V**iew

Chapter 4

HR	**H**igh **R**esolution
NPA	**N**on-**P**edestrian **A**rea
CPU	**C**entral **P**rocessing **U**nit
WHO	**W**orld **H**ealth **O**rganization
FOV	**F**ield **O**f **V**iew
ICF	**I**ntegrate **C**hannel **F**eatures
ACF	**A**ggregated **C**hannel **F**eatures
LDCF	**L**ocally **D**ecorrelated **C**hannel **F**eatures
SCF	**S**quares**C**hn**F**trs
ELM	**E**xtreme **L**earning **M**achine

ANN **A**rtificial **N**eural **N**etwork

Chapter 5

AVOR **A**utonomous **V**ehicle **O**bject **R**ecognition
AUC **A**rea **U**nder **C**urve

Chapter 6

PL **P**seudo **L**iDAR

Chapter 7

SNN **S**piking **N**eural **Network**

Contents

List of Figures

List of Tables

Part I

Introduction and State of the Art

Chapter 1

Introduction

1.1 Background

Autonomous vehicles (AVs) present a technological revolution opportunity for countries to achieve social progress. For regions, AVs have the ability to accelerate the circular flow of production factors. For individuals, AVs will reduce the potential for accidents caused by driver failure. Despite there are concerns about AVs replacing human drivers, such as job losses and privacy protection, we should be aware that the risks and opportunities co-exist. If we look at history, the increase in production usually brings prosperity to all mankind. Therefore, we should embrace the changes and guide AV technology to a path that benefits everyone.

The Society of Automotive Engineers (SAE) establishes a taxonomy level model[1] for autonomous vehicles. There are six levels of driving automation system from L0 to L5, aiming to progressively reduce driver involvement. The tasks of different automation levels are defined as shown in Table 1.1. It is worth noting that this is a blueprint from the technical perspective. In practical scenarios, drivers will not perform as expected. As in the Waymo 2021 Safety Report[2], when they tested a L3 AV, the human drivers over-trusted the technology and were not monitoring the roadway carefully. Therefore, in our opinion, AV should be classified into 3 levels. Level A0: No Driving automation. Level A1: Advanced Driver Assistance Systems. Level A2: Fully autonomous driving system. Most autonomous driving companies, such as Waymo, Apollo and lyft, are currently trying to tackle L4 level AV.

Table 1.1: Different levels of driving automation system

SAE	Description	Ours
L0	No Driving Automation	A0
L1	Driver Assistance	A1
L2	Partial Driving Automation	
L3	Conditional Driving Automation	A2
L4	High Driving Automation	
L5	Full Driving Automation	

In order to achieve vehicle autonomy, the priority is to acquire environmental information through multiple sensors. As shown in Figure 1.1, the sensor equipment typically includes IMU, GPS, ultrasound, Radar, camera and LiDAR. On this basis, several key modules of AV are constructed which are localization, perception,

[1] https://www.sae.org/standards/content/j3016_202104/
[2] https://waymo.com/safety/

prediction and reaction. Localization module finds out where the vehicle is in the map. Perception module obtains the position and movement information of the objects around the vehicle. Prediction module estimates the behavior of other objects in order to find a possible path. Reaction module determine the trajectory, speed and steering maneuvers required to proceed along the route.

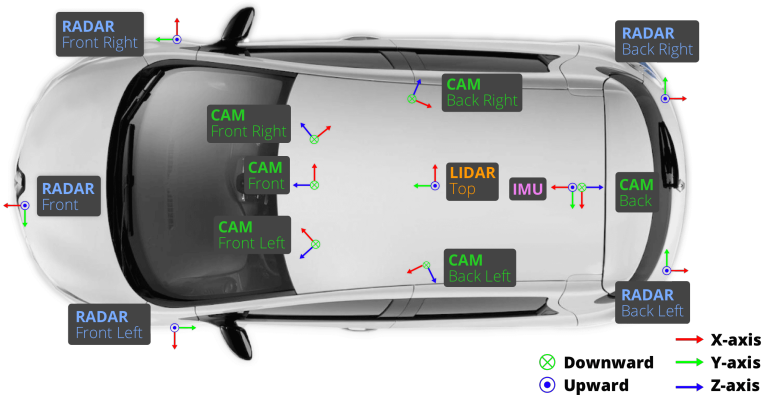


Figure 1.1: Multi-modal sensor equipment of nuScenes [Caesar et al., 2020].

The perception module is fundamental to the prediction and reaction modules. There are several subtasks in perception module such as detection, classification and segmentation. The topic of this thesis focuses on object detection. The purpose of object detection is to locate object's position in the given data modalities. The detection result is normally a rectangular box or cubic box enclosing the object.

Over the last decade, object detection accuracy has been progressed significantly. Benefiting from deep learning, the detector learns to recognize patterns directly from data. A deep learning object detection network can be considered as two components. The first is feature extraction and the second is task regression. Efficient feature extraction and increasing annotated data are two essential backbones for designing deep learning networks.

Deep learning is one of the machine learning methods that stacks multiple nonlinear layers in depth to simulate the shape of features in high dimensions. Since the introduction of AlexNet [Krizhevsky et al., 2017] in 2012, convolutional neural networks (CNN) have become one of the representative deep learning networks. The 2D convolution layer of CNN achieves outstanding performance in image-based 2D object detection task. However, for AVs, the location information of the object in spatial is required. The absence of depth information in the image modality makes it difficult to retrieve precisely in discrete pixels. As a result, 3D object detection

based on image modality suffers from low accuracy.

For this reason, most AVs are equipped with LiDAR to provide accurate depth information in points form. The point cloud referred to as LiDAR modality is completely different compared to the image modality. These points are discrete, disordered and sparse. The networks designed for image modality are inappropriate and inefficient for LiDAR modality data. Over the last five years, extensive research has been conducted on performing efficient and accurate 3D detection in LiDAR modality. In the process of improving 3D detection accuracy, studies on fusing multiple modalities are becoming increasingly relevant.

Image modality is enriched with semantic information like color. LiDAR modality provides accurate contours of object in spatial. The fusion method is intended to aggregate the specialties of both modalities to achieve robust and accurate detection. Contrary to intuition, the effective fusion of these two modalities is still an open challenge.

This thesis intends to explore in detail multi-modal fusion for 3D object detection. For this purpose, the deep learning based object detection algorithms are first summarized in **Chapter 2**. In the following, 3D object detection is investigated in four aspects.

Chapter 3 Improvement of evaluation metric average precision. Inspired by the paper [Simonelli et al., 2019], the evaluation metric average precision (AP) for object detection is investigated. The AP calculation method is usually an N-point interpolation method, such as the 11-point interpolation method employed in KITTI [Geiger et al., 2012]. Yet, Existing N-point interpolation methods could produce severe errors. These errors lead to average precision distortion, which makes it impossible to accurately evaluate the performance of the model. For this reason, the enhanced N-point interpolation method is proposed. First, the interpolation point position is changed to the middle interpolation. Second, Dynamic selection of parameters for calculating the area of the interpolation interval. After the improvement, the AP distortion during evaluation is reduced by 90%. The main contributions of this chapter are:

(1) The risk and reason of AP distortion is exposed and explored.

(2) The enhanced N-point interpolation method is introduced for fast and accurate evaluation of object detection model.

Chapter 4 Unified visual model for dataset optimization. Increasing the resolution of the image does help in detecting small or distant objects, as more informa-

tion can be captured. In exchange, the background region is also expanded which strains the operation of the system. To reduce dataset redundancy, we build a visual model based on pedestrian targets. This visual model, called Non-Pedestrian Area Estimation (NPAE), is used to compute the non-pedestrian area that are uniformly present in all images of the dataset. The performance of NPAE is evaluated by a 2D object detector. The experimental results show that the NPAE algorithm has a significant improvement in detection speed, while the detection accuracy is well preserved. The main contributions of this chapter are:

(1) A unified optimization path for the large number of images in a dataset is proposed.

(2) Accordingly, a visual model called NPAE is proposed to reduce data redundancy and retain useful information.

Chapter 5 Late multi-modal fusion 3D object detection. We find that a large number of false positives (FP) exist in the 3D object detection results based on LiDAR modality only. However, the projection of these FP proposals onto the image plane is easily eliminated by a classifier. Therefore, we propose a cross-modal verification (CMV) model for reducing 3D object detection false positives. In this fusion approach, color and texture information in image modality and the precisely depth information in LiDAR modality are combined. Result shows that more than 50% of false positives in 3D object detection proposals are eliminated. The main contributions of this chapter are:

(1) The classification task is introduced in multi-modal fusion for 3D object detection.

(2) The individual classification module can be integrated with any 3D object detector.

Chapter 6 Asynchronous multi-modal fusion for 3D object detection. Autonomous vehicles are often equipped with multiple modal sensors in order to have a robust 3D detection of the environment. All existing fusion methods require that the data is already well synchronised. However, as asynchronous frames have been discarded, the sampling frequency of synchronous data will inevitably decrease. The slow detection frequency may lead to collisions in autonomous vehicles due to insufficient reaction time. The AF3D is proposed for the unified detection on both synchronous and asynchronous frames. For the synchronous frames, the LiDAR modality is selected for 3D object detection. For asynchronous frames with only image modality, we transform the image modality to LiDAR modality. The

asynchronous fusion based on scene flow employs images of asynchronous and synchronous frames to estimate the 3D motion of each pixel. The point cloud of the synchronous frame is then relocated to generate the point cloud of the asynchronous frame, according to the 3D motion. As a result, the same 3D object detector for synchronous frames could apply for asynchronous frames. The main contributions of this chapter are:

(1) The scene flow is introduced into multi-modal fusion for 3D object detection.

(2) An asynchronous fusion paradigm is proposed to aggregate synchronous and asynchronous frames.

(3) The 3D object detection accuracy in image modality is improved 9.4% by asynchronous multi-modal fusion.

Finally, **Chapter 7** concludes the thesis and provides perspectives on future work.

Chapter 2

Background

Contents

This chapter presents the background of 2D and 3D object detection. The state-of-the-art object detection methods are based on deep learning which is firstly introduced. We also bring a brief summary of the metric for object detection.

2.1 Deep Learning

2.1.1 The Develop of Deep Learning

Deep learning is one of the most rapidly expanding technical branches of machine learning. The explosive growth of data quantity and the continuous improvement of computing power allow for enrichment of deep learning applications. In recent years, deep learning has demonstrated excellent performance in autonomous driving, natural language processing and medical image processing.

Traditional machine learning algorithms require experts to construct features from raw data [Dalal and Triggs, 2005, Lowe, 2004]. The quality of the features directly determine the effectiveness of subsequent tasks. Besides, the features extracted by the experts are domain specific and difficult to integrate in other domains. Conversely, the most successful design of deep learning is the automatic feature extraction. It makes deep learning an easily interdisciplinary tool that can be quickly applied in different domains.

The neural network of deep learning simulate the operation of the brain. The front layers in the neural network extract low-level features, and the back-end layers construct high-level features based on multiple low-level features. Eventually, the output layer forms concepts based on the automatically extracted features.

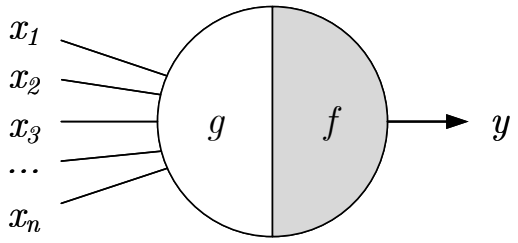


Figure 2.1: McCulloch-Pitts model.

The beginning of deep learning is the McCulloch-Pitts (MCP) model [McCulloch and Pitts, 1943] proposed for simulating neurons as shown in Figure 2.1. g takes an input and performs an aggregation. f is the threshold logic which

makes a decision based on the previous aggregated value. MCP demonstrates the reasoning procedure, yet unable to learn.

Based on MCP model, Frank Rosenblatt proposes the Perceptron architecture [Rosenblatt, 1958]. The Perceptron is a linear classification model. A loss function is introduced and minimized using the gradient descent method to build the perceptron model. The architecture of Perceptron is shown in Figure 2.2.

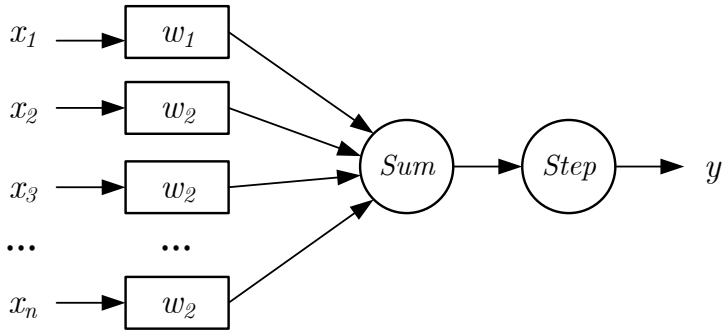


Figure 2.2: Perceptron model.

With the invention of back propagation, the training of a deep learning model has become possible. Subsequently, the prototype of Convolutional Neural Network, Neocognitron [Fukushima and Miyake, 1982] is introduced. In this context, LeNet [LeCun et al., 1998] is produced and achieves reasonably good accuracy on the handwritten number recognition task. LeNet is a convolutional neural networks and uses back propagation to train the network as shown in Figure 2.3.

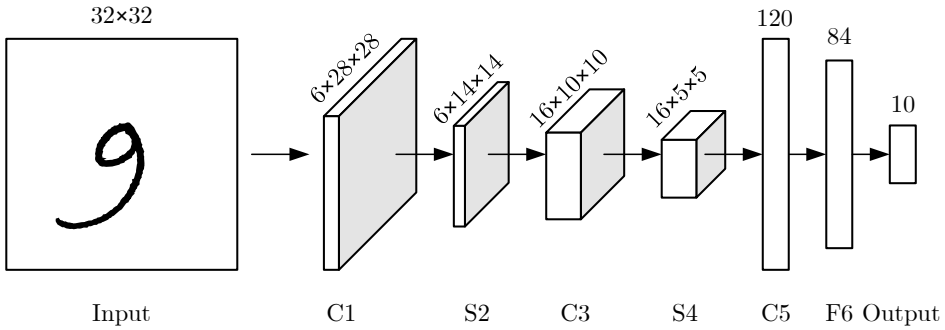


Figure 2.3: LeNet-5 model.

Thereafter deep learning has witnessed an explosive period of progress. Among the various deep learning networks, convolutional neural network is the most brightly growing. Many fundamental neural network architectures and concepts have been created successively.

In 2012, AlexNet [Krizhevsky et al., 2017] is introduced to ImageNet competition. AlexNet has 600 million parameters with a total of 650,000 neurons located in 5 convolutional layers. Dropout is adopted to randomly ignore a portion of neurons during training to avoid overfitting the model. By using *ReLU* as the activation function of CNN, the gradient dispersion problem of *Sigmoid* when the network goes deeper is overcome. Moreover, AlexNet uses graphics processing unit (GPU) to accelerate the training of CNN and increases the number of training samples through data augmentation.

In 2014, VGG-19 [Simonyan and Zisserman, 2014] demonstrates that increasing the depth of the network has influence on the final performance of the network. Each layer of the neural network uses the output of the previous layer to further extract more complex features. VGG-19 uses the 3×3 convolutional kernel instead of the larger convolutional kernel in AlexNet. In this way, the depth of the convolutional layer is increased to ensure that the neural network learning more complex patterns.

When the depth of a neural network increases to a certain level its performance decreases instead. Too many parameters make the neural network have a tendency to overfit and make training difficult due to gradient dispersion. GoogLeNet [Szegedy et al., 2015] provides a sparse network structure using 1×1 convolutional kernels to downscale the input channels and reduce the number of model parameters. It combines the feature maps extracted from the multi-scale convolutional and pooling layers into the next layer which improves the model generalization.

In 2015, ResNet [He et al., 2016] provided an architecture to construct a neural network with more than 1000 layers. To address the performance degradation of deep neural networks, ResNet uses a residual module that allows convolutional layers to be connected across layers. Also, Batch Normalization (BN) layer is used to solve the gradient vanishing problem.

In 2017, the Transformer [Vaswani et al., 2017] based entirely on the attention module is proposed. Transformer is composed of self-Attention and Feed Forward Neural Network only. A trainable neural network can be built by stacking the Transformer without any CNN. It was first used in Natural Language Processing (NLP) and then extended to traditional CNN applications such as object detection.

As the study of 2D image-based deep learning becomes fruitful, the research of more complex 3D environments has gradually attracted attention. In autonomous driving, the data modalities for 3D task are usually LiDAR, image and Radar. Based

on different modalities, many new deep learning methods have been developed, and we will describe them in detail below.

2.2 2D Object detection

2.2.1 Convolutional Neural Network

Convolutional neural network (CNN) is composed of neurons with learnable weights and biases in each layer. Every neuron performs dot product on the input data. It could learn the weights directly from data without manually designing rules. CNN is a powerful tool to reveal patterns in images for 2D object detection task. In this section we present the components of a basic CNN. Figure 2.4 shows the difference between MLP and CNN.

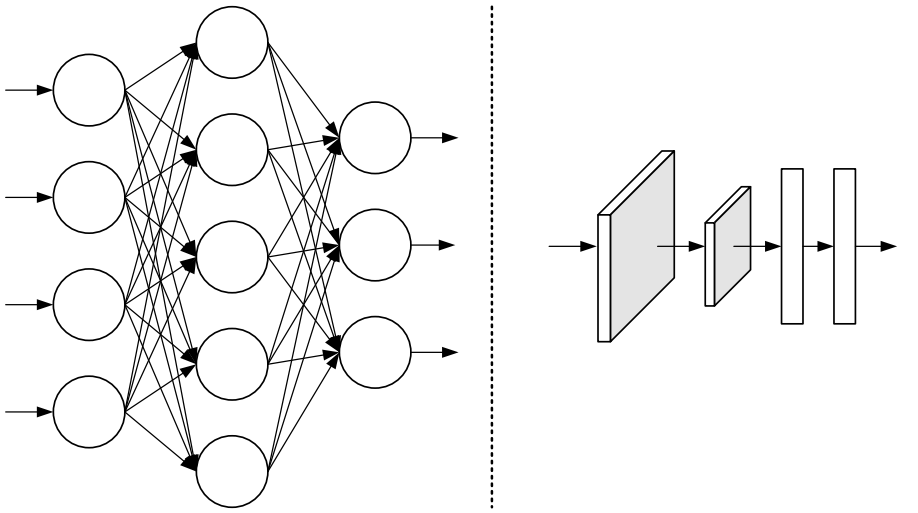
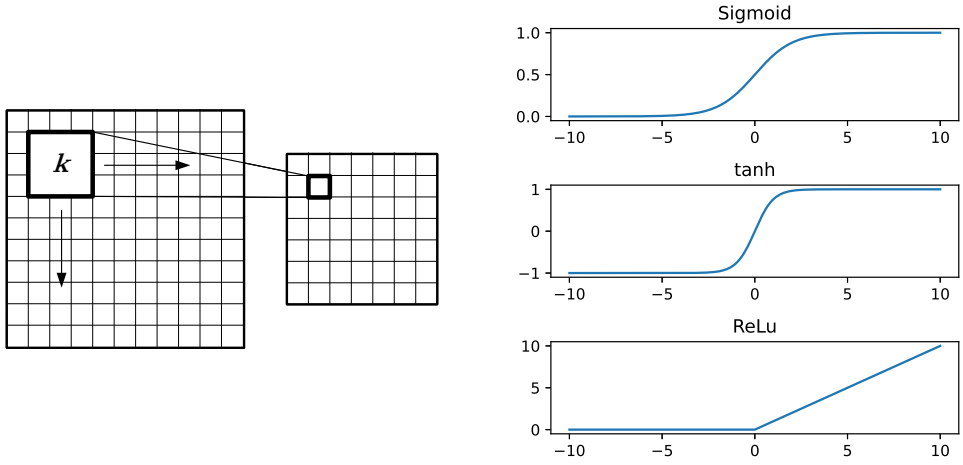


Figure 2.4: Compare of MLP and CNN.

Generally, a CNN network is composed of an input layer, one or multiple hidden layers and an output layer from the perspective of data flow. These three functional layers are constructed from convolutional layers, activation layers, pooling layers and fully connected layers.

- **Convolution layer**

The convolution layer uses a convolution kernel to extract the feature maps and applies an activation function to add non-linearity. This allows the neural network



(a) Convolution kernel (b) Activation function

Figure 2.5: Convolution layer.

to approximate any nonlinear function. The two modules of convolution layer are shown in Figure 2.5

Convolution kernel. Convolution kernel is a linear operation used for feature extraction. A kernel is an array of 3×3 typically in size. The process of convolution is the element-wise product of the convolution kernel with the input tensor at all positions. The tensor obtained after convolution is called a feature map and represents a feature pattern. In this case, the convolution layer usually contains multiple convolution kernels and outputs a feature map with multiple channels.

Activation function. The most commonly used nonlinear activation function is rectified linear unit (ReLU). The mathematical expression for ReLU is $f(x) = max(0, x)$. ReLU is a nonlinear function that allows the neural network to fit a nonlinear model. Compared with sigmoid and tanh, the derivative of ReLU is better to solve, which makes the network training faster.

- **Pooling layer**

The pooling layer is a downsampling of the feature map. There are two commonly adopted pooling layers which are max pooling and average pooling as shown in Figure 2.6. Taking the 2×2 max pooling as an example, the pooling formula is $p(x) = max(x)$. The pooling operation could provide several benefits. 1) Enables feature extraction with a certain degree of resistance to distortions and shifts. 2) It

allows the same size convolution kernels to have different receptive fields. 3) Reduce the size of feature map and thus reduce the computational amount.

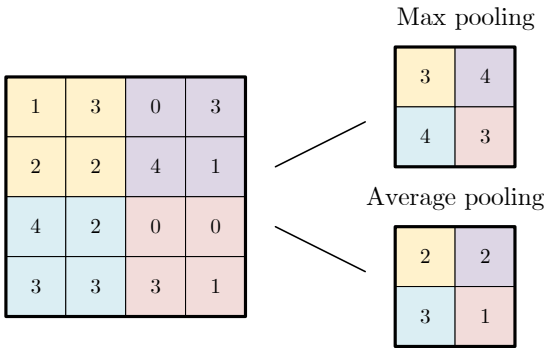


Figure 2.6: Pooling layer.

- **Fully connected layer**

The feature maps need to be transformed into the final output format such as a vector. Here, the fully connected layer is connected to each position in all feature maps as shown in Figure 2.7. The flattened 1D array feature is computed by $f(x) = \mathrm{W}x + b$. Then, similar to the convolution layer, a nonlinear function is followed. The number of neurons in the last fully connected layer should be designed according to the application, such as the number of classes in the classification task.

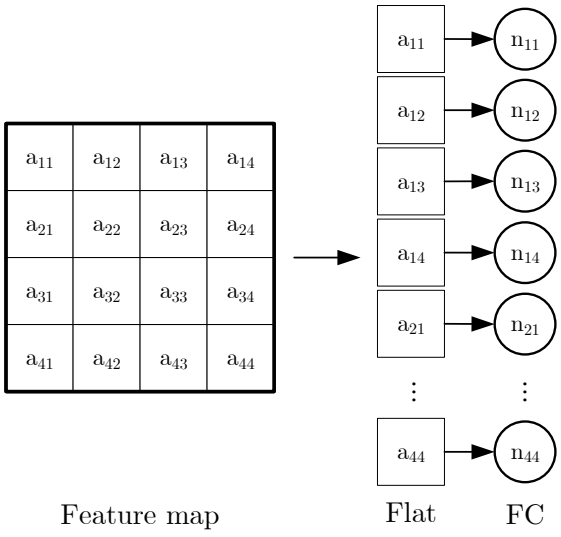


Figure 2.7: Fully connected layer.

- **Backward Propagation**

Training a CNN network is to find the proper parameters for convolution kernels and fully connected weights. The aim of the training is to make the inference result the same as the given ground truth. The optimization algorithm to adjust those parameters is back propagation. The amplitude of the parameter adjustment is determined by the loss obtained through forward propagation.

A commonly used back propagation algorithm is gradient descent. The gradient of the loss function provides the fastest direction to reduce the loss. Each learnable parameter is updated with an arbitrary step size determined based on a hyperparameter called learning rate. The gradient is a partial derivative of the loss with respect to each learnable parameter.

2.2.2 Method

Object detection is an important task for computers to perceive the environment. The detection task includes distinguishing object classes and locating object instances. After years of development, the detection methods have gradually developed from traditional methods to deep learning-based methods. Traditional detectors, such as SIFT [Ng and Henikoff, 2003] and HOG [Wang et al., 2009], rely on hand-crafted features. In contrast, CNN-based deep learning object detectors learn feature extraction directly from the data. As a result, a new level of accuracy in object detection has been achieved.

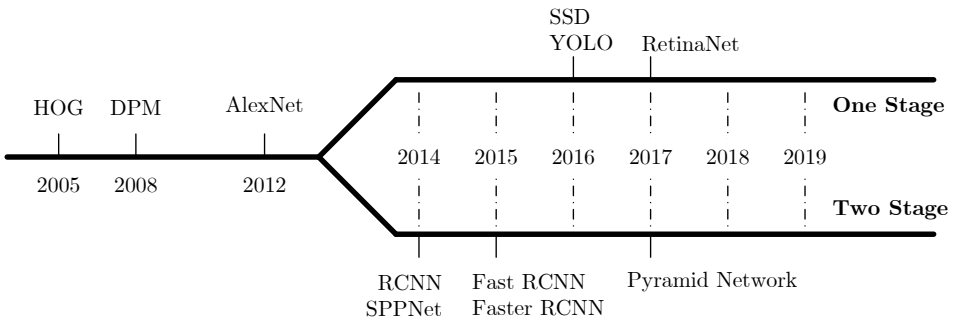


Figure 2.8: Summary of 2D object detection

In the era of deep learning, object detectors can be divided into two categories: 1) Region proposal based two-stage detector. This type of detector first generates region proposals and then classifies each proposal. 2) Regression based one-stage

detector. This kind of detector considers the object detection as a regression problem on the bbox coordinates. The develop of object detection is shown in Figure 2.8.

Region Proposal based Two-stage Detectors

- RCNN

In 2014 Ross Girshick proposed the Regions with CNN features (RCNN) [Girshick et al., 2014] with an impressive mean average precision (mAP) improvement from 33.7% (DPM [Girshick et al., 2015]) to 58.5%.

There are three modules that cooperate to implement object detection in RCNN. 1) Region proposal generation. RCNN runs selective search [Uijlings et al., 2013] to generate around 2000 region proposals on a single image. All region proposals are adjusted to the fixed resolution 227×227 for the next module. 2) Feature extraction by CNN. RCNN employs CNN to extract features instead of hand-crafting them as in traditional methods. Each region proposal is finally represented by a 4096-dimensional feature vector. 3) Classification. After previous processing, the features are sent to the classifier. In this way, the object detection problem is transformed into a classification problem. The classifier could be a support vector machine (SVM) [Hearst et al., 1998] or any other algorithm. The scored features are filtered by non-maximum suppression (NMS) to select the best bounding box proposals.

Despite the significant improvement in detection accuracy, RCNN still has defects. The feature extraction of 2000 region proposals with a resolution of 227×227 in an image obviously consumes a considerable amount of time. The large amount of overlap in these region proposals results in redundant feature extraction.

- SPPNet

He Kaiming proposed the spatial pyramid pooling neural network (SPPNet) [He et al., 2015] based on spatial pyramid matching (SPM) [Lazebnik et al., 2006]. SPPNet removes the region proposal generation and extracts features directly from the image. At the last convolution layer, the pooling layer is replaced by the spatial pyramid pooling (SPP) layer. The feature maps acquired from the last convolution layer are divided by 1×1, 2×2 and 4×4 grids. In this way, a total of 21 feature bins are obtained representing the fine and coarse scale partitioning of the image. In order to feed the following fully connected layer which requires fixed-length features, these bins are sent to the max pooling, which yields 21 feature representations.

SPPNet saves a lot of reasoning time by avoiding repeated computation of features. Besides, the input image is not restricted to a fixed size. This means that the object will not be deformed by wrapping. As a result, SPPNet achieves more than 20 times faster than R-CNN without sacrificing any detection accuracy.

- Fast RCNN

Although SPPNet has been improved compared with RCNN, it still has some defects. The training is multistage including feature extraction, classifier training and bounding box regression training. In addition, only the fully connected layer is fine-tuned and the weights of the convolution layer are preserved. Fast RCNN [Girshick, 2015] is proposed in 2015 to confront the above problems.

There are several influential developments in Fast RCNN. 1) RoI pooling layer. The input to Fast RCNN is an image and its region proposals are generated by selective search. To avoid repeated feature extraction in RCNN, Fast RCNN uses CNN to retrieve features in one step. Similar to SPPNet, the feature maps from the last convolution layer are sent to a regions of interest (RoI) pooling layer. The RoI pooling layer is a special case of the SPP layer, which has only one pyramid level. The partition grid is 7×7. As a result, a fixed-length feature vector is able to be processed by following fully connected layers.

2) Multitask loss. Before Fast RCNN, the detection task was built on top of classification task by adding a bounding box regression module after the classifier. In Fast RCNN, classification and detection are designed as two parallel tasks which share the same convolutional backbone network. For this reason, Fast RCNN is a multitask network which requires a multitask loss to indicate the training direction.

For the classification task, Fast RCNN outputs a discrete probability distribution $p = (p_0, \ldots, p_{\mathrm{K}})$ for every RoI over $\mathrm{K}+1$ categories including K object classes and a background class. The classifier has been replaced by softmax instead of SVM. Each RoI is labeled with a ground truth class u and the loss for classification is defined in Equation 2.1.

$$\mathrm{L}_{cls}(p, u) = -\log p_u \tag{2.1}$$

For the detection task, Fast RCNN regresses the offsets of the RoI to locate the bounding box. The offsets $t^k = (t^k_x, t^k_y, t^k_w, t^k_h)$ and $k \leq \mathrm{K}$ where K is the number of object classes. Therefore, the loss for detection task is defined in Equation 2.

$$\mathrm{L}_{loc}(t^u, v) = \sum_{i \in x,y,w,h} \mathrm{smooth}_{\mathrm{L}_1}(t_i^u - v_i) \tag{2.2}$$

where the prediction for class u is $t_u = (t_x^u, t_y^u, t_w^u, t_h^u)$ and the bounding box regression target $v = (v_x, v_y, v_w, v_h)$. The L_1 loss is defined as below.

$$\mathrm{smooth}_{\mathrm{L}_1}(x) = \begin{cases} 0.5x^2 & \text{if} |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases} \tag{2.3}$$

Fast RCNN combines the two losses with a balance hyperparameter λ to get the multitask loss in Equation 2.4.

$$\mathrm{L}(p, u, t^u, v) = \mathrm{L}_{cls}(p, u) + \lambda \lfloor u \geq 1 \rfloor \mathrm{L}_{loc}(t^u, v) \tag{2.4}$$

3) End-to-End training. Since Fast RCNN integrates classification and detection into one model, the multitask loss L is derived from one forward propagation. Then the critical node that needs to be addressed is the RoI pooling layer for back propagation as shown in Equation 2.5.

$$\frac{\partial \mathrm{L}}{\partial x_i} = \sum_r \sum_j [i = i * (r, j)] \frac{\partial \mathrm{L}}{\partial y_{rj}} \tag{2.5}$$

where $x_i \in \mathbb{R}$ is the ith activation input to the RoI pooling layer. y_{rj} is the jth output from the rth RoI. $i * (r, j) = \mathrm{argmax}_{i' \in \mathrm{R}(r,j)} x_{i'}$ and $\mathrm{R}(r, j)$ is the index set of partitioned and pooled y_{rj}. Calculating the back propagation gradient of RoI Pooling can be treated as calculating the gradient of max pooling for each segmented block separately, and then summing up all the gradients. In this way the weights of the convolution layers and fully connected layers can be adjusted according to CNN back propagation.

Through extensive improvements, Fast RCNN achieves significant increase in detection accuracy. The mean average precision (mAP) on VOC07 dataset is 70.0%, which outperforms the previous detectors. Moreover, the bounding box regression and CNN feature extraction for object detection task become an end-to-end integrated network. The inference speed of Fast RCNN is more than 200 times faster than that of RCNN, which makes real-time detection possible.

- Faster RCNN

Fast RCNN is inadequate in that the region proposals were still generated

separately rather than integrated into the network. The following Faster RCNN [Ren et al., 2015] made up for this deficit by introducing the region proposal network (RPN). After the final feature extraction from convolution layer, the RPN is inserted. This small fully convolution network slides over the feature map. At each location, the RPN determines whether there are objects within 9 predefined anchors. There is another sibling channel in RPN to acquire bounding box regression. The classification features and bounding box regression features are used to crop proposals from the feature maps of CNN. Then these region proposals are sent to RoI pooling layer and further classified and regressed.

Faster RCNN achieves a 73.2% mAP on VOC07 dataset. Furthermore, the processing speed is 17 fps with ZFNet [Zeiler and Fergus, 2014] which had been the fastest deep neural network based detector yet.

- Feature Pyramid Network

Scale variation is one of the most difficult issues in object detection. Small objects in the image are easily ignored by the detector. The reason is that after the convolution and pooling layers, the semantic information of small objects almost disappears in the high-level feature maps. In order to confront such problems, based on Faster RCNN [Ren et al., 2015], SSD [Liu et al., 2016], ResNet [He et al., 2016] and Feature Image Pyramid [Lin et al., 2017a], the Feature Pyramid Networks (FPN) [Lin et al., 2017a] is proposed.

The feature maps from different layers of the CNN have different scales. More pixels are occupied by an object in the low-level feature map than in the high-level one. This implies that small objects are more easily detected in the low-level feature maps. Therefore, it is possible to perform object detection on all feature maps of different layers. However, it requires that the lower convolution layer must have very powerful feature extraction capabilities. Therefore FPN upsamples the high-level features and then fuses them with the low-level features. By fusing the layers in this way, the low-level feature maps are also rich in semantic information.

Using FPN in a basic Faster RCNN network, it achieves state-of-the-art object detection result on the COCO dataset [Lin et al., 2014].

Regression based One-stage Detectors

- You Only Look Once (YOLO)

YOLO [Redmon et al., 2016] is a revolutionary object detection framework which abolishes the region proposal procedure. The bounding box regression is conducted directly by the convolution layers shared with the classification. In other words, detection and classification can be regressed by CNN in one shot.

YOLO network divides the input image of VOC dataset with a resolution of 448×448 into a 7×7 grid. Each grid cell is responsible for the detection and classification of the objects within it. To achieve this goal, YOLO scales down the image size by 64 times after 24 layers of convolution. As a result, the size of final feature maps after convolution layers is 7×7 with 1024 channels. Then through 2 fully connected layers, the result tensor with a shape of $7 \times 7 \times 30$ is acquired. There are 49 tensors with the shape of $1 \times 1 \times 30$, mapped to the input image. For each grid cell, there are 20 classes and 2 bounding boxes with confidence scores need to be regressed.

YOLO is the first one-stage CNN-based detector. Since there is no region proposal and classification stage, the detection speed reaches an amazing 155 fps with a mAP score 52.7% on VOC07 dataset.

- Single Shot MultiBox Detector (SSD)

YOLO divides the image into a 7×7 grid, which presumes the presence of only one object in each grid cell. The small objects in the same grid cell will inevitably be ignored, resulting in a decrease in detection accuracy. SSD [Liu et al., 2016] is proposed to address this issue. There are several important novel designs in SSD.

1) Pyramid feature map for detection. There are six channels of feature map which are sent to perform detection. Small objects are more likely to be detected on low-level feature maps. Large objects are more likely to be detected in high-level feature maps. 2) Preset anchor boxes. SSD presets several anchor box shapes and performs bounding box regression at each location on the feature map. Among all convolution layers, for the conv4_3, conv10_2 and conv11_2, there are 4 default anchor boxes associated. And for conv7, conv8_2 and conv9_2, there are 6 default anchor boxes associated. Therefore, there are 8732 bounding box proposals for regression. 3) Detection with fully convolution layer. Unlike Yolo which finally uses fully connected layers, SSD directly uses convolution on different feature maps to extract detection results. For a feature map with a shape of $m \times n \times p$, SSD uses a convolution kernel of $3 \times 3 * p$ to get the detection proposals.

SSD significantly improves the detection accuracy of a one-stage detector. The mAP on the VOC07 dataset is 76.8%, while the detection speed could reach at 59 fps

in the fast version.

- RetinaNet

RetinaNet [Lin et al., 2017b] is a one-stage detector that integrates several outstanding structures. The feature pyramid network (FPN) is adopted as backbone to extract feature maps. The preset anchors are designed with reference to region proposal network (RPN). There are also a classification subnet and a box regression subnet as promised by the one-stage detector.

The most important progress in RetinaNet is the introduction of focal loss. The focal loss is proposed to solve the positive and negative sample imbalance problem. These designs allow the RetinaNet to have the accuracy of the two-stage detector while maintaining the speed of the one-stage detector.

- CornerNet

Anchor is a key component of object detection whether one-stage or two-stage. However, the size and number of anchor box are both hyperparameters which need to be adjusted repeatedly for different situations. In addition, many anchors are preset on the feature map to prevent missing objects. However, the positive samples are much less than the negative samples. The problem of imbalanced samples may lead to difficult training of the network.

For this reason, the CornerNet [Law and Deng, 2018] is proposed, which is the first anchor free object detector. Anchor free means that there is no preset anchor in the network. CornerNet is inspired by instance segmentation which labels pixels into different classes and groups these pixels into an individual object. Similar to this concept, CornerNet considers object detection as finding the upper left and lower right corners of the bounding box. The experiment results show that CornerNet outperforms state-of-the-art one-stage detector of the time.

- CenterNet

CenterNet [Duan et al., 2019] is an improved anchor free detector. Unlike CornerNet, CenterNet model an object as the center point of its bounding box. And the bounding box width and height are inferred from the center keypoint.

For CenterNet, an image is processed by a fully convolutional network and a heatmap is obtained. Based on the heatmap, the keypoint estimation to find center points and regresses to all other properties.

2.3 3D Object detection

3D object detection is a key component for autonomous vehicle environment perception. Compared to 2D environment, 3D environment increases the depth information and makes object detection more complex. In order to locate an object in 3D space, the cubic bounding box has the format $[x_c, y_c, z_c, l, w, h, \theta, c]$. The (x_c, y_c, z_c) is the center point coordinate of the object. The l, w, h is the length, width and height of the object, separately. θ is the heading angle. c is the class of the object.

Camera and LiDAR are two common sampling data sensors used for 3D object detection. The image modality acquired by the camera can be directly used directly for 3D object detection, regardless of the lack of depth information. The data sampled by LiDAR is in point cloud format which is unordered and sparse. As a result, 2D feature extraction methods designed for dense image modality cannot be directly applied to 3D detection. There are three types of 3D object detection methods, depending on the modality involved.

2.3.1 LiDAR-based architecture

- Point-based

A point cloud is a series of points that can be stored in a matrix with the shape of $N \times 3$ where N is the number of points. The points stored adjacent to each other are unrelated in space. In other word, the point cloud is invariant under transformations and is unordered. Points in regions in 3D space jointly depict object contours.

To address the problem in point cloud feature extraction, PointNet [Qi et al., 2017a] is proposed. First, the input is aligned by multiplying it with the transformation matrix in T-Net. This operation ensures the invariance of the model to the spatial transformation such as affine transformation and rigid transformation. Then after several multi-layer perceptron (MLP), there is another T-Net for feature alignment. Finally, the most important operation is to apply the symmetry function to deal with the unordered features. The symmetry function takes n vectors as input and outputs a new vector that is invariant to the input order. In PointNet, the symmetry function is max pooling. The architecture of PointNet is shown in Figure 2.9.

The ideas presented in PointNet are instructive and have a lot of potential. In response to the shortcomings of the PointNet design, specific improvements are

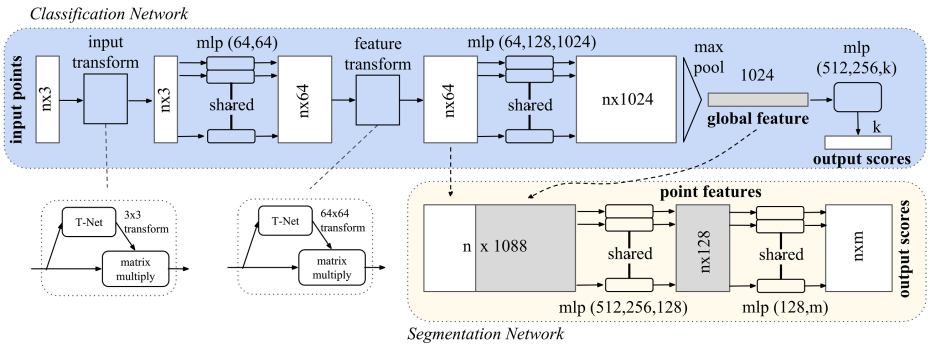


Figure 2.9: Architecture of PointNet [Qi et al., 2017a]

made in PointNet++ [Qi et al., 2017b] as shown in Figure 2.10.

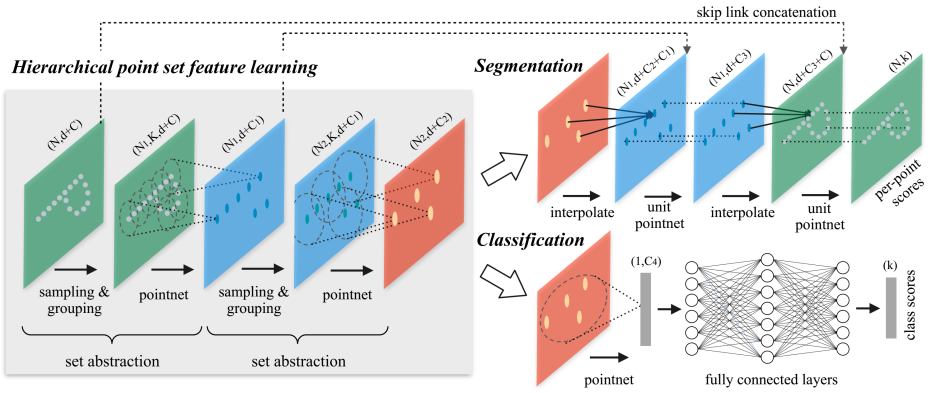


Figure 2.10: Architecture of PointNet++ [Qi et al., 2017b]

1) Hierarchical feature learning. In PointNet, the global features are derived from the max pooling, where a lot information is discarded. Instead, PointNet++ adopts several set abstraction to downsample the feature maps. Thus local-global features of different sizes are obtained in different levels.

2) Farthest point sampling (FPS). Since there are enormous number of points and the points are not uniformly distributed, it is very difficult to extract features. FPS is used for selecting a subset of points to represent the whole input points. To achieve this goal, FPS first adds an initial point randomly to the selected point set. Then, using the initial point as the target point, find the point farthest from it and add that point to the selected point set. Update the target point to the newly found point and iteratively search for the rest. The search will end until the number of sampling points meets the requirement.

The PointNet series of methods provides a pioneering solution for deep learn-

ing to extract features directly from point clouds. Modules such as FPS and SA have been applied in many 3D object detection methods such as PointRCNN [Shi et al., 2019]. The Figure 2.11 shows a common architecture of point-based 3D object detector.

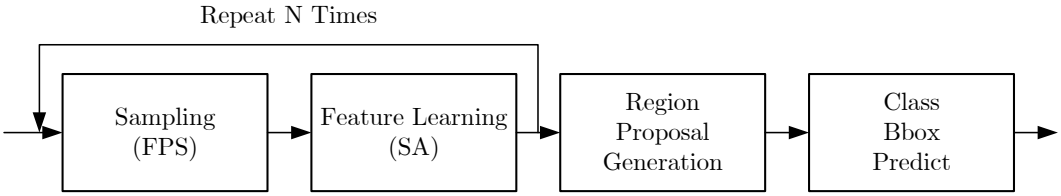


Figure 2.11: Common architecture of Point-based 3D object detector

The PointNet++ is adopted by PointRCNN as the backbone network. For the generation of bounding box proposals, PointRCNN introduces the bin-based 3D bounding box generation module. After the foreground point segmentation module, the separated foreground points are used for regression to generate bounding box proposals. Bin-based 3d box generation is a coarse-to-fine method that could accelerate training.

Sampling is a critical challenge for point-based methods. Too many sampling points will affect the calculation efficiency, while too few of them will reduce the accuracy. In addition, since the furthest point sampling is intrinsically a sequential algorithm, it is a bottleneck in reference time.

- Voxel-based

The voxel-based methods attempt to convert the sparse point cloud into a dense format. A voxel is considered as a pixel in a 3D space. Each voxel has the same size and is allocated with a discrete coordinate. The general architecture of a voxel-based 3D detector is shown in Figure 2.12.

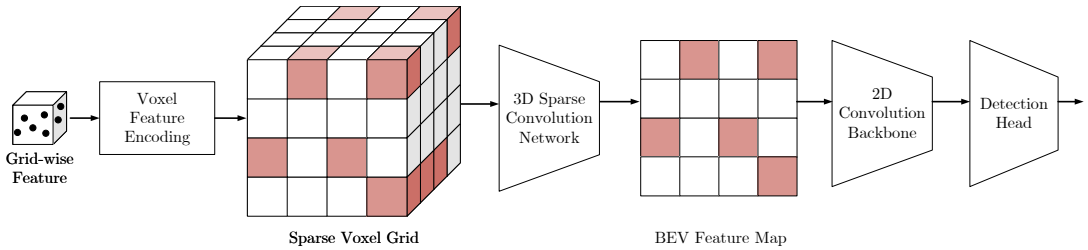


Figure 2.12: Architecture of a voxel-based 3D detector.

VoxelNet [Zhou and Tuzel, 2018] is a pioneering work to apply voxel to the field of 3D object detection. There are three modules in the VoxelNet which are 1) Feature learning network. 2) Convolutional middle layers. 3) Region proposal network. The whole architecture is similar to a traditional 2D two-stage object detector. Among them, the feature learning network is designed for point cloud feature extraction based on voxel.

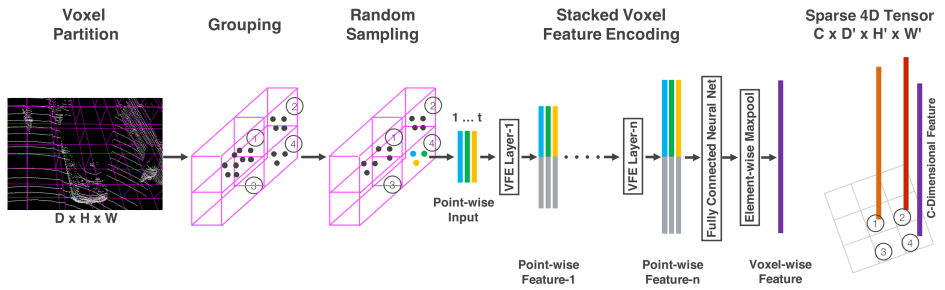


Figure 2.13: Feature learning network of VoxelNet [Zhou and Tuzel, 2018].

1) Feature learning network. There are five steps in the feature learning network as shown in Figure 2.13. First, the 3D space is separated by the voxel partition. Followed by the grouping operation, which places the points into the voxels they belong to. Then, the number of points is reduced using random sampling. Since the 3D space is voxel-partitioned, downsampling can be accelerated in a random way. The next is the most important design, named voxel feature encoding (VFE) layer.

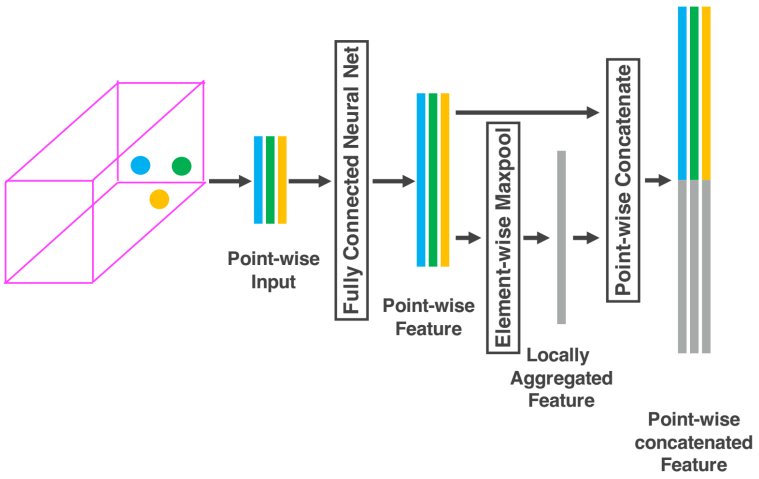


Figure 2.14: Voxel feature encoding of VoxelNet [Zhou and Tuzel, 2018]

As shown in Figure 2.14, there are three points in the voxel, and point-wise feature are extracted by the fully connected network. After that, locally aggregated

feature is obtained by using maxpool. Then this local feature is concatenated to each point-wise feature. The point-wise concatenated feature is then ready for the next VFE layer. Finally, through an element-wise maxpool, a representative feature named voxel-wise feature is acquired. The voxel features are represented by a 4-dimensional sparse tensor, which is used to reduce memory and computational consumption during back propagation.

2) Convolutional middle layers. VoxelNet adopts 3D convolution layer which could severely limit system performance. SECOND [Yan et al., 2018] proposes a sparse convolutional middle layer to improve the computationally intensive problem. This layer has become a common component of voxel-based detectors.

3) Region proposal network. After the convolutional middle layers, the features have been extracted. Then comes the regression of the bounding box. The 2D convolution layer could be applied directly since the feature maps are well transformed. The region proposal network regresses the 7-dimensional vector representation of the bounding box.

There is a special type of voxel partitioning named pillar. Pillar can be considered as a stacking of voxels along the depth direction. The architecture of the pillar-based 3D object detector is shown in Figure 2.15.

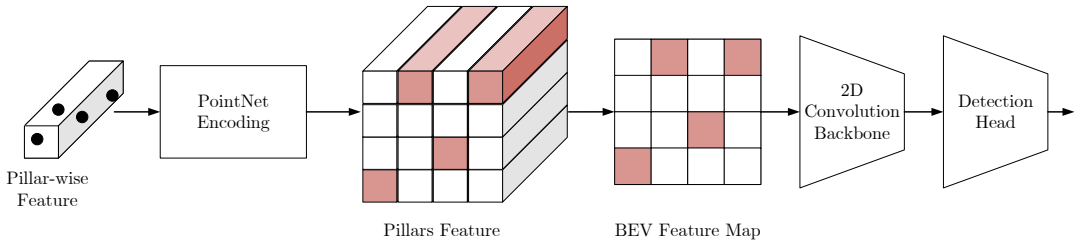


Figure 2.15: Architecture of the pillar-based 3D object detector.

PointPillars [Lang et al., 2019] is a representative pillar-based 3d object detector. Unlike the Voxel-based method, PointPillars converts point clouds into pseudo-images by partitioning pillars and extracting features. In this way, the feature extraction is accelerated and the advanced 2D object detection module can be applied.

The most critical impact of voxel-based methods on detection performance is the partitioning of voxels. Discrete voxels inevitably lose some structural information. But at the same time, it also brings an increase in detection efficiency. For example, PointPillars is capable of performing 3D object detection at a frequency of up to 105Hz. Balancing voxel size and detection performance remains an open challenge.

- Point-voxel based methods

Since the point-based and voxel-based methods have their own advantages, a point-voxel based fusion method is proposed. PVRCNN [Shi et al., 2020a] provides an effective point-voxel based architecture for 3D object detection.

There are three steps in PVRCNN. 1) Voxel to 3D proposals. PVRCNN utilizes a series of 3D sparse convolution to gradually convert the point cloud into 3D features. Then the 3D features are converted into 2D bird-view feature maps and 3D proposals are generated. 2) Voxel to keypoint. In parallel with the voxel-based method the furthest point sampling algorithm is adopted to sample a group of keypoint. Then, PVRCNN introduces a voxel set abstraction (VSA) module to encode the multi-scale features from voxel-based method into keypoints. 3) Keypoint to grid. The RoI-grid pooling module is proposed to aggregate the keypoint features to the RoI-grid points. In this way, it is capable to learn features for fine-grained proposal refinement.

In conclusion, the point-voxel based algorithm sacrifices inference time to obtain detection performance improvement compared to the voxel-based algorithm.

2.3.2 Image-based

Compared to LiDAR, the camera is an ideal sensor that is cheap and easy to acquire. Many studies attempt to use images for direct 3D object detection. However, various benchmark results show that the image-based method still exists a great gap with the LiDAR-based method. The reason for this gap is the error in recovering the depth information from the image. Two subdivisions of image-based methods have been developed to compensate for defects in images : 1) Result-lifting based. 2) Feature-lifting based.

- Result-lifting based

The method in the result-lifting subdivision decomposes the 3D object detection task into two tasks: 2D object detection and depth estimation.

GS3D [Li et al., 2019a] is a representative 3D detector based on monocular images. The 2D object detector is first applied to predict 2D bounding boxes. Then, a coarse cuboid is derived from each 2D bounding box. Finally, the coarse cuboid is refined by using a classification formulation with quality aware loss.

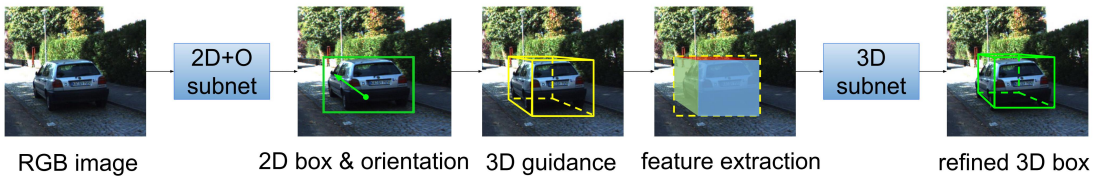


Figure 2.16: Architecture of GS3D [Li et al., 2019a].

With disparity, better depth estimates can be obtained from stereo images. Based on this property, stereo R-CNN [Li et al., 2019b] performs region proposal network on a pair of images. In order to migrate 2D proposals to 3D space, more features need to be extracted from the stereo images. One branch applies RoI align and fully-connected layers to extract semantic information. The other branch predicts four 3D semantic keypoints which indicate four corners at the bottom of the 3D bounding box. Finally, all above information is sent to the 3D box estimation module for final results.

- Feature-lifting based

The methods in this subdivision expect to lift features from 2D to 3D by generating point clouds or learning depth distribution. Pseudo-LiDAR [Wang et al., 2019] is one of the representative methods.

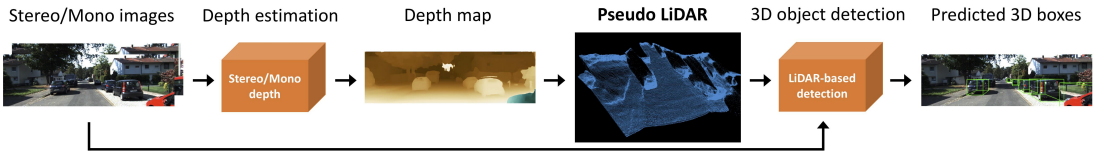


Figure 2.17: Architecture of Pseudo LiDAR [Wang et al., 2019].

The architecture of Pseudo-LiDAR is shown in Figure 2.17. A depth estimator first generates a depth map based on the input image. According to the back-projection function, depth and pixel position can yield points in 3D space, which is called pseudo-LiDAR. The modality of the data is transferred from the image to the point cloud. As a result, any available LiDAR-based 3D detector could be applied.

2.3.3 Fusion-based

Both image and point cloud modalities have their own unique characteristics. Image modality could provide color semantic information. While point cloud could

accurately describe the contours of objects in 3D space. Ideally, the fusion of both modalities should improve the accuracy of object detection. Yet, the fusion-based approach do not outperform the LiDAR-based approach. Therefore, effectively combining multiple modalities for 3D object detection remains an open challenge.

Due to the superior performance achieved by LiDAR-based detectors, existing fusion-based approaches attempt to integrate image features in different parts of LiDAR-based detectors. According to this principle, fusion-based methods can be divided into three categories: 1) Early fusion. 2) Middle fusion. 3) Late fusion.

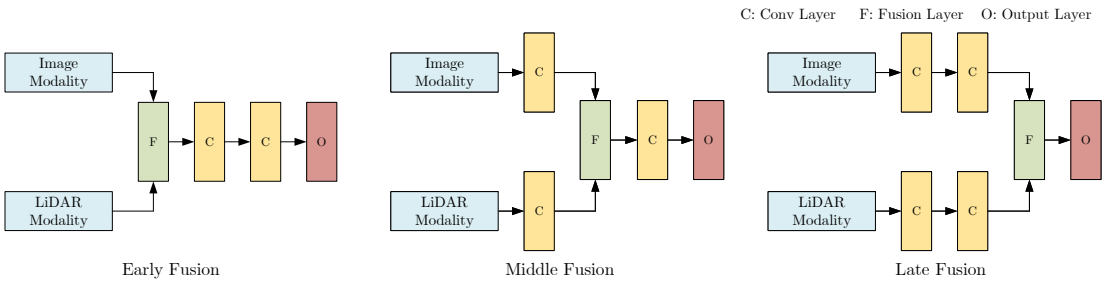


Figure 2.18: Different types of multi-modal fusion.

- Early fusion

In early fusion way, the raw data or preprocessed data are fused. Algorithms explore multi-modal data in the early stages to learn common features. The advantage of this early fusion paradigm is that the network does not need to be designed separately for the different modalities. Meanwhile, data alignment and sensor replacement can have a serious impact on the fusion system.

- Middle fusion

In addition to early fusion of data, middle fusion of features is also an option. Through feature concatenation, middle fusion could take place in the backbone network or the region proposal generation. For middle fusion, the effective aggregation of features is the critical point.

- Late fusion

The late fusion scheme combines decision outputs of each domain specific network of a sensing modality. It focus on the aggregation of different modality outputs.

Late fusion avoids the complex design of middle fusion. As a price, the features of the different modalities are not associated and may drop the enriched information.

It is worth noting that different stages of fusion can co-exist to help the network make better use of multi-modal features. For example, both middle and late fusion are performed in the MV3D [Chen et al., 2017] network. MV3D first projects the point cloud into the bird eye view and front view. Then, with the original image, these three images are treated like an image-based 3D detection procedure. After RoI pooling, features are repeatedly fused until the final result is obtained.

A representative network for early fusion is MVXNet [Sindagi et al., 2019]. In order to learn the interaction between different modalities, point cloud is projected to image plane for feature extraction. Then the features of the projected points are fused with the image features. Meanwhile, the other path adopts a LiDAR-based method to extract point cloud features. After the point cloud features and image features are concatenated, the VFE structure of VoxelNet is applied.

2.4 Metric

Average precision (AP) is a commonly adopted metric to evaluate model detection performance in most benchmarks and competitions. The AP score reflects the composite of precision and recall of the detector on one class of object. For multiple classes, the mean average precision (mAP) could represent the comprehensive detection performance of the detector. The process of calculating AP for both 2D and 3D object detectors is to first calculate the intersection over union (IoU), then plot the precision-recall (PR) curve, and finally calculate the AP score.

From the detector, the bounding box (BBox) proposals are obtained. Compared tools ground truth BBox, these proposals could be judged as correct or incorrect. A common way to measure the similarity between the proposals and ground truth is the intersection over union (IoU). The method collects the overlapping area and the union area of a proposal BBox and its corresponding ground truth BBox and the IoU score is the division of these two areas.

It is not necessary for the proposal BBox perfectly matches the ground truth BBox. There is a threshold t which is used to classify the correctness. If the proposal BBox IoU score is greater than t, then it is considered to be correct. The opposite is considered to be incorrect.

The set of proposal BBox and ground truth BBox can be divided into three cat-

egories: 1) True positive (TP), which indicates the correct proposal BBox. 2) False positive (FP), which indicates the incorrect proposal BBox. 3) False negative (FN), which indicates the unmatched ground truth BBox.

All proposal BBox are sorted according to the confidence score for drawing PR curve. Since the proposal BBox have been given correctness and sorted, it is easy to know the value of TP, FP and FN in each position of the sorted array. A precision score p at each recall r position is obtained, where $p = \mathrm{TP}/(\mathrm{TP}+\mathrm{FP})$ and $r = \mathrm{TP}/(\mathrm{TP}+\mathrm{FN})$. By aggregating the precision values at every position of proposal BBox, the PR curve can be plotted. The average precision score is the area under PR curve.

The area under the curve is approximated by using several interpolated rectangles. The number of interpolation points N is flexible, and more points brings higher accuracy. The method of calculating the area under the PR curve by interpolating at each recall is the all-point interpolation method. Taking the 11-point interpolation method as an example, the AP is calculated as follows:

$$\mathrm{AP}_{11} = \frac{1}{11} \sum_{\mathrm{R}\in 0,0.1,\ldots,0.9,1.0} \mathrm{P}_{it}(\mathrm{R}) \tag{2.6}$$

where

$$\mathrm{P}_{it}(\mathrm{R}) = \max_{\mathrm{R}':\mathrm{R}'\geq \mathrm{R}} \mathrm{P}(\mathrm{R}') \tag{2.7}$$

The mAP which is used to measure the detection accuracy over all classes C is acquired by:

$$\mathrm{mAP} = \frac{1}{\mathrm{C}} \sum_{i=1}^{\mathrm{C}} \mathrm{AP}_i \tag{2.8}$$

2.5 Conclusion

In this chapter, we presented the background of the object detection. Deep learning is one of the fundamental techniques for many computer vision tasks. It learns to extract features directly from the raw data. This provides an important basis for the development of object detection. Two types of detection architecture have been developed for 2D object detection using deep learning. After the features extracted by CNN, one architecture directly regresses the object location, the other one generates region proposals which are then classified. 3D object detection is an extension of 2D object detection. With a feature extractor designed for 3D data, it converts the

3D object detection to 2D object detection. At the end, we presented the metric to evaluate the object detection, the average precision.

Part II

Contributions

Chapter 3

An Enhanced N-Point Interpolation Method to Eliminate Average Precision Distortion

Contents

3.1 Motivation

Existing N-point interpolation methods generate large errors in the average precision calculation for object detection. These errors lead to average precision distortion, which makes it impossible to accurately evaluate the performance of the model. We investigate the reason for the average precision distortion and propose an enhanced N-point interpolation method. These improvements are based on the N-point interpolation method and can be summarized in two parts: 1) The interpolation point position is changed to the middle interpolation. 2) Dynamic selection of parameters for calculating the area of the interpolation interval. Experiments verify the existence of severe average precision distortion in the N-point interpolation method. Furthermore, the proposed enhanced N-point interpolation method reduces the average precision distortion by more than 90% to only 0.04%. In this way, the enhanced N-point interpolation method is able to replace the all-point interpolation method for fast and accurate evaluation of object detection model.

3.2 Problem

3.2.1 Introduction

Object detection is an essential backbone component of many computer vision applications, such as autonomous driving [Zhang et al., 2021b, Zhang et al., 2021a], medical diagnosis [Lung et al., 2021], and video understanding [Han et al., 2020]. In order to determine the performance of various detection models, an accurate evaluation metric is fundamental. For object detection task, the average precision is a widely adopted metric that considers both precision and recall for the final score.

According to the definition [Zhu, 2004], the average precision is the area under the precision-recall curve, which can be calculated using the all-point interpolation method. The all-point interpolation method is a definite integral method, where it is accurate, but computationally intensive. [Everingham et al., 2010, Geiger et al., 2012, Lin et al., 2014] have adopted another kind of fast algorithm for average precision calculation called the N-point interpolation method. It requires only N calculations to obtain an approximate average precision value, usually N can be chosen as 11 or 101.

Inevitably, the increase in speed brings a decrease in accuracy. There is an er-

ror in the average precision calculated by the N-point interpolation method, which we call the average precision distortion. The amount of average precision distortion can be determined by calculating the average precision difference between the N-point interpolation method and the all-point interpolation method. Average precision distortion is common in the N-point interpolation method, and it causes chaotic evaluation results on the model performance. In [Simonelli et al., 2019], the authors describe the average precision distortion that exists at the first interpolation point and leads to an overestimation of the model. In addition, average precision distortion may cause several anomalies as follows. A1) A better performing detector may have a lower average precision score instead. A2) Two different performing detectors may have the same average precision value. A3) The average precision of two similar performing detectors may be too far apart.

Notably, the average precision distortion is a systematic error, which can be reduced by improving the existing methods. Therefore, we propose an enhanced N-point interpolation method based on the N-point interpolation method by changing the interpolation point position and the area calculation parameters of the interpolation interval. The experimental results show that the average precision distortion of the enhanced N-point interpolation method is reduced by more than 90%. In particular, the enhanced 40-point interpolation method has an average precision distortion of only 0.04% on the KITTI 3D object detection dataset [Geiger et al., 2012].

3.2.2 Related works

The precision and recall of the detection are the foundation for calculating the average precision. Along with true positives (TP), false positives (FP) and false negatives (FN), precision and recall can be defined by Equation 3.1.

$$\begin{aligned} Precision &= \frac{N_{tp}}{N_{tp}+N_{fp}} = \frac{N_{tp}}{N_{dt}} \\ Recall &= \frac{N_{tp}}{N_{tp}+N_{fn}} = \frac{N_{tp}}{N_{gt}} \end{aligned} \tag{3.1}$$

Where N_{tp} means the number of true positives. N_{fp} means the number of false positives. N_{fn} means the number of false negatives. N_{dt} means the number of detection results and N_{gt} means the number of ground truths.

The Precision-Recall (PR) curve can be drawn based on the recall rate and the

corresponding precision value. The average precision score is the area under the PR curve. There are two type of methods to calculate the Area Under the Curve (AUC): the all-point interpolation method and the N-point interpolation method.

3.2.3 All-point interpolation method

The recall is discrete in the range $[0,1]$ with an interval of $1/\mathrm{N}_{gt}$. According to the principle of definite integration, the area under the PR curve can be obtained by summing the area of the rectangle at each recall. Therefore, the all-point interpolation method is defined in Equation 3.2.

$$\begin{aligned} \mathrm{AP}_{\mathrm{all}} &= \frac{1}{\mathrm{N}_{gt}} \sum_{\mathrm{R} \in \mathbb{R}_{all}} \mathrm{P}_{max}(\mathrm{R}) \\ \mathrm{P}_{max}(\mathrm{R}) &= \max_{\mathrm{R}':\mathrm{R}' \geq \mathrm{R}} \mathrm{P}(\mathrm{R}') \end{aligned} \tag{3.2}$$

Where the set of interpolation points for the all-point interpolation method is $\mathbb{R}_{all} = \{1/\mathrm{N}_{gt}, 2/\mathrm{N}_{gt}, ..., \mathrm{N}_{tp}/\mathrm{N}_{gt}\}$. The N_{tp} is the number of true positives. $\mathrm{P}(\mathrm{R}')$ is the precision at the interpolation point R'. The condition $\mathrm{R}' : \mathrm{R}' \geq \mathrm{R}$ indicates that the $\mathrm{P}_{max}(\mathrm{R})$ is the maximum precision among the interpolation point R and subsequent interpolation points.

The all-point interpolation method has an accurate calculation of the area under the PR curve. However, its computational effort increases significantly with the amount of data. The variety of task modes and parameter choices available in object detection further complicates the situation. Moreover, considering that the benchmark and competition platform have to deal with a large number of detection models, a lower evaluation speed is unacceptable. Therefore, it is essential to find a fast and accurate average precision calculation method that could replace the all-point interpolation method.

3.2.4 N-point interpolation method

To simplify the all-point interpolation method, the N-point interpolation method divides the entire recall interval $[0,1]$ into N regions. The area under the PR curve for each interpolation interval is approximated by the area of the rectangle at the interpolation point. The definition of the N-point interpolation method is presented in Equation 3.3.

$$\begin{aligned} \mathrm{AP_N} &= \frac{1}{\mathrm{N}} \sum_{\mathrm{R} \in \mathbb{R}_\mathrm{N}} \mathrm{P}_{max}(\mathrm{R}) \\ \mathrm{P}_{max}(\mathrm{R}) &= \max_{\mathrm{R}':\mathrm{R}' \geq \mathrm{R}} \mathrm{P}(\mathrm{R}') \end{aligned} \tag{3.3}$$

Where $\mathbb{R}_\mathrm{N}$ is the set of N interpolation points. Specifically, $\mathbb{R}_{11} = \{0, 1/10, 2/10, ..., 10/10\}$ for 11-point interpolation method [Geiger et al., 2012]. $\mathbb{R}_{40} = \{1/40, 2/40, ..., 40/40\}$ for 40-point interpolation method [Simonelli et al., 2019]. $\mathbb{R}_{101} = \{0, 1/100, 2/100, ..., 100/100\}$ for 101-point interpolation method [Lin et al., 2014].

Paper [Simonelli et al., 2019] identifies an anomaly in the average precision score of the 11-point interpolation method at the first interpolation point. After a thorough study, we realize that this anomaly phenomenon exists in all kinds of N-point interpolation method and seriously affects the accuracy for model performance evaluation. The most essential presentation of the problem is the error in the average precision calculated by the N-point interpolation method compared to the all-point interpolation method. We refer to this error as average precision distortion, which has been fully analyzed in the next chapter.

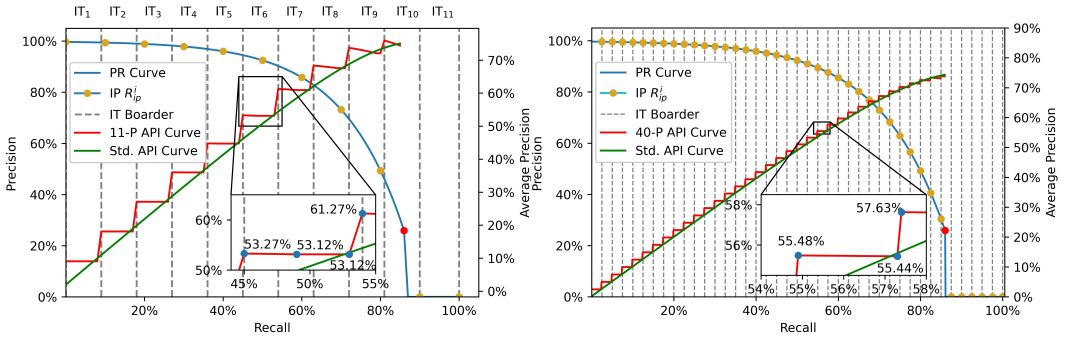
3.3 Average Precision Distortion

The N-point interpolation method calculates only the rectangular area of N interpolation intervals under the PR curve. There exists a certain error between the rectangular area and the area under the curve. Besides, the N-point interpolation method ignores the differences between the interpolation intervals, resulting in average precision values that contain significant errors. These are the vital causes of average precision distortion.

To observe the average precision distortion, we propose a novel average precision increase (API) curve, as shown in Figure 3.1(a) and 3.1(b). The API curve is obtained by calculating the area under the PR curve segment using the N-point interpolation method continuously as the recall increases. By comparing with the exact area under the PR curve segment referred to as the standard API curve, the average precision distortion of N-point interpolation method can be determined.

The procedure to formulate the API curve is as follows. For a certain PR curve, we intercept a segment $[0, \mathrm{R}_n]$ of this curve, where the recall $\mathrm{R}_n = n/\mathrm{N}_{gt} \times 100\%$, $n = 0, 1, 2, ..., \mathrm{N}_{gt}$. The area under the PR curve segment is then calculated by using the

N-point interpolation method. Eventually a series of average precision scores can be obtained, forming the API curve. By replacing the N-point interpolation method with the all-point interpolation method, the standard API curve can be acquired.



(a) (b)

Figure 3.1: (a) The average precision increase (API) curve of 11-point interpolation method (11-P API Curve). (b) The average precision increase curve (API) of 40-point interpolation method (40-P API Curve). Both the 11-P and 40-P API curves have various anomalies compared to the standard (Std.) API curve. IP is interpolation point. IT is interpolation interval.

$$\mathrm{PR}_{ex} = \begin{cases} e^{5.4\frac{n}{\mathrm{N}_{tp}}-5.7}+1 & ,n=0,1,\ldots,\mathrm{N}_{tp} \\ 0 & ,n=\mathrm{N}_{tp}+1,\ldots,\mathrm{N}_{gt} \end{cases} \tag{3.4}$$

To simplify the quantitative analysis, PR_{ex} in Equation 3.4 is chosen as a representative PR curve. $\mathrm{PR}_{ex} = e^{5.4n/\mathrm{N}_{tp}-5.7}+1$ on the recall $[0, \mathrm{N}_{tp}/\mathrm{N}_{gt}]$. $\mathrm{PR}_{ex} = 0$ on the recall $[\mathrm{N}_{tp}/\mathrm{N}_{gt}, 100\%]$. The recall is acquired as n/N_{gt}. The number of ground truth $\mathrm{N}_{gt} = 100$ and the number of true positives $\mathrm{N}_{tp} = 86$. The PR_{ex} curve is shown as the blue line in Figure 3.1(a) and (b). It is important to note that the above parameters are chosen to facilitate the presence of average precision distortion. The same conclusion can be obtained by replacing the parameters for the investigation.

For the 11-point interpolation method, the process to obtain the API curve of PR_{ex} is as follows. First, the average precision value ap_1 of the PR_{ex} curve segment $[0, 1\%]$ for $n = 1$ is computed by the 11-point interpolation method. Then, for the PR_{ex} curve segment $[0, 2\%]$ where $n = 2$, the ap_2 can be acquired. Repeat to increase n until ap_{86} is obtained, where $n = \mathrm{N}_{tp}$. We obtained a series of average precision values $[ap_1, ap_2, \ldots, ap_{86}]$ which is the API curve obtained by increasing n.

In Figure 3.1(a), the red line is the API curve of the 11-point interpolation method and the green line is the standard API curve. The difference between the two curves at the same recall rate indicates the presence of average precision distortion. The

further the distance between the two curves is, the more severe the average precision distortion is. It can be clearly seen that the average precision distortion is numerically variable and always exists.

Four points of the 11-P API curve in Figure 3.1(a) are chosen for comparative analysis which are R_{45}, R_{49}, R_{53} and R_{54}. Among them, R_{45} is the first point of the interpolation interval IT6. R_{49} is the interpolation point position of the IT6. R_{53} is the last point of the IT6. R_{54} is the first point of the IT7. The average precision values of these four points can be regarded as obtained by calculating four similar PR curves using the 11-point interpolation method. Assuming the four PR curves corresponding to four detectors, Det1, Det2, Det3 and Det4. Then the performance ranking of the four detectors can be determined based on the difference in area under the curve as Det1 < Det2 < Det3 < Det4. However, multiple anomalies exist on the API curve that differ from the above conclusion.

A1) Compare the point R_{45} and R_{49}, the $AP_{N|R_{45}} = 53.27\% > AP_{N|R_{49}} = 53.12\%$, which means the performance of Det1 is better than Det2. It is contrary to the real situation, indicating that better performing detectors may instead obtain lower average precision scores.

A2) Compare the point R_{49} and R_{53}, the $AP_{N|R_{49}} = 53.12\% = AP_{N|R_{53}}$, which implies that D$et$2 has the same performance as Det3. This indicates that more correct detection results may not lead to a change in average precision.

A3) For the point R_{53} and R_{54}, despite the increase of only one true positive, the average precision increased by $AP_{N|R_{54}} - AP_{N|R_{53}} = 8.15\%$. However, for average precision values within the same interpolation interval, the difference between R_{45} and R_{53} is only 0.15%. This indicates that average precision distortion can cause two detectors with similar performance to obtain average precision scores that differ significantly.

Figure 3.1(b) shows the API curve of the 40-point interpolation method based on the same PR curve in Figure 3.1(a). Similar to the 11-point interpolation method, the 40-point interpolation method also suffers from average precision distortion problem, which is a common defect of the N-point interpolation method and needs to be improved.

The main reason for the average precision distortion is that the N-point interpolation method does not calculate the area of the last valid interpolation interval correctly. The recall range of the PR curve is $\nu = [0, R_e]$. The valid interpolation interval refers to the interpolation interval $[R_l, R_r] \cap \nu$ is not null. As shown in Figure

3.1(a), from IT1 to IT10 are the valid interpolation intervals and the IT10 is the last valid interpolation interval.

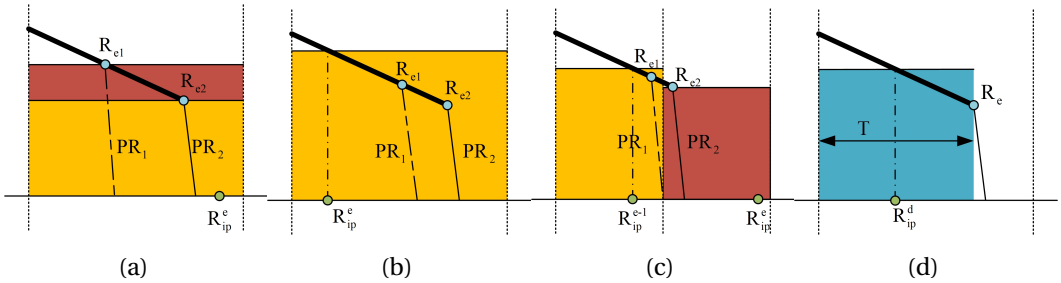


Figure 3.2: (a) The area under the PR_2 curve is larger yet the average precision score (The orange area) is lower. (b) The area under the PR_1 and PR_2 curves are different while the average precision scores are the same. (c) The difference in area under the PR_1 and PR_2 curves is small but the difference in average precision scores is large. (d) Dynamically selects the area calculation parameters for the last valid interpolation interval. R_{e1} and R_{e2} are the last point of the PR_1 and PR_2, respectively. R^e_{ip} and R^{e-1}_{ip} are the interpolation points. The valid interval width T is the actual mapping length of the PR curve segment on the Recall axis. The dynamic interpolation point for the last interpolation interval R^d_{ip} is the interpolation point located in the middle of T. Better viewed in color.

Calculating the area S_i of each valid interpolation interval requires the use of the precision value at the interpolation point $P_{max}(R^i_{ip})$, as shown in Equation 3.5.

$$S_i = \frac{1}{N} \times P_{max}(R^i_{ip}) \tag{3.5}$$

Where the R^i_{ip} is the i^{th} interpolation point. The $P_{max}(R^i_{ip})$ the maximum precision of the interpolation point R^i_{ip} and subsequent interpolation points. N is the total number of interpolation points. The Equation 3.5 is calculating the area of a rectangle with width 1/N and height $P_{max}(R^i_{ip})$.

Unlike other valid interpolation intervals, there are two possibilities for determining the value of the interpolation point in the last valid interpolation interval. The determination of which case it belongs to is based on the relationship between the last point in PR curve $R_e = N_{tp}/N_{gt}$ and the interpolation point R^e_{ip} of the last valid interpolation interval.

- **(1)** $R_e < R^e_{ip}$. In this case, the N-point interpolation method makes $R^e_{ip} = R_e$. As shown in Figure 3.2(a), the $R^e_{ip} = R_{e1}$ for the PR_1 and $R^e_{ip} = R_{e2}$ for the PR_2.

- **(2)** $R_e \geq R^e_{ip}$. In this case, there is a valid value of R^e_{ip} as shown in Figure 3.2(b).

Both of these two cases of determining R^e_{ip} could cause severe average precision distortion. Suppose there are two PR curves, PR_1 and PR_2, as shown in Figure 3.2(a). R_{e1} and R_{e2} are the last points on the PR_1 and PR_2 curves, respectively. The segment $[0, R_{e1}]$ of PR_2 curve is exactly the same as PR_1 curve. Therefore, it should be $S(PR_2) > S(PR_1)$ when comparing the area S under the curve. However, since $R_{e1}, R_{e2} < R^e_{ip}$, then for the PR_1 curve, $R^e_{ip} = R_{e1}$ and for the PR_2 curve, $R^e_{ip} = R_{e2}$. As a result, $AP_{PR_2} < AP_{PR_1}$. This is the opposite of the actual situation. This proves the case described above, that a better detector yet has a lower average precision score.

For the $R_e \geq R^e_{ip}$ shown in Figure 3.2(b), $P(R^e_{ip})$ has a valid value. It implies that for the PR_1 and PR_2 curves, the average precision scores will be the same. This is the reason why more correct detection results do not lead to a change in average precision.

Figure 3.2(c) shows a special case where PR_2 has one more true positive compared to PR_1, that leads to one more interpolation interval. For this reason, even though the actual difference between the area under the PR_1 and PR_2 curves is slight, there will be a significant difference between AP_{PR_1} and AP_{PR_2}. This is the trigger for the sudden leap in the API curve near the interpolation interval in Figure 3.1(a) and (b).

3.4 Enhanced N-point interpolation method

To address the average precision distortion, we propose an enhanced N-point interpolation method to achieve better accuracy and stability in the average precision calculation. The enhanced N-point interpolation method includes the following two improvements based on the N-point interpolation method.

3.4.1 Middle interpolation point position

The selection of the interpolation point position is very important because its precision value is the height of the interpolation interval rectangle. Since every interpolation interval is different, the ideal interpolation point position inevitably differs as well. Therefore, we need to find the representative interpolation point position so that the error of area calculation for all interpolation intervals is minimized.

The positions of the interpolation points that minimize the average precision distortion μ for each interpolation interval of the PR_{ex} curve are shown in Table

3.1. The average precision calculation method is the 11-point interpolation method. The average precision distortion $\mu = AP_A - AP_{11}$. AP_A is the average precision calculated by the all-point interpolation method. AP_11 is the average precision calculated by the 11-point interpolation method. $PR_{ex} = e^{5.4n/N_{tp}-5.7} + 1$, $n = 0, 1, \ldots, N_{tp}$. The number of true positive is $N_{tp} = 840$. The number of ground truth is $N_{gt} = 1000$. The recall can be obtained by $n/N_{gt} \times 100\%$.

Table 3.1: The positions of the interpolation points that minimize the average precision distortion for each interpolation interval of the PR_{ex} curve. $PR_{ex} = e^{5.4n/N_{tp}-5.7} + 1$, $n = 0, 1, \ldots, N_{tp}$. The number of true positive is $N_{tp} = 840$. The number of ground truth is $N_{gt} = 1000$.IT means the interpolation interval. IP means interpolation point position. The preferred interpolation point positions are near the middle.

IT	**IT1**	**IT2**	**IT3**	**IT4**	**IT5**	**IT6**	**IT7**	**IT8**	**IT9**
$\mathbf{IP}_{best}$	90	27	37	42	45	46	47	48	48

The PR_{ex} curve is divided into 11 interpolation intervals by the 11-point interpolation method. The interpolation intervals 1 to 10 are the valid interpolation intervals. The interpolation interval 10 is the last valid interpolation interval. Here, we focus on the interpolation point position for the interpolation interval 1 to 9.

Each interpolation interval has 90 points. Every point is used as an interpolation point to calculate the average precision distortions. The interpolation point position with the lowest average precision distortion among the 90 values is selected to form the Table 3.1.

We also inspect the interpolation point position for the 40-point interpolation method. There are 34 valid interpolation intervals. Each interpolation interval has 25 points and the middle of the interpolation interval is 13. For the interpolation intervals 1 to 33, the average preferred interpolation point position is 13 which is the middle of the interpolation interval.

Table 3.2: The average precision distortion μ caused by different interpolation point position. Middle interpolation (Middle Interp) has the best performance.

	Slide Interp	End Interp	**Middle Interp**
11-P μ	1.0107%	3.1941%	**-0.1620%**
40-P μ	0.2900%	0.8925%	**-0.0111%**

The average precision distortion caused by different interpolation point positions is shown in Table 3.2. The average precision distortion generated by calculating the last valid interpolation interval is not included in the table. The

results demonstrate that the middle interpolation has the least average precision distortion. Three different interpolation point positions are shown in Figure 3.3. The slide interpolation is used for the 11-point interpolation method [Everingham et al., 2010, Geiger et al., 2012] and 101-point interpolation method [Lin et al., 2014]. The end interpolation is used for the 40-point interpolation method [Simonelli et al., 2019]. We propose to use the middle interpolation in the enhanced N-point interpolation method.

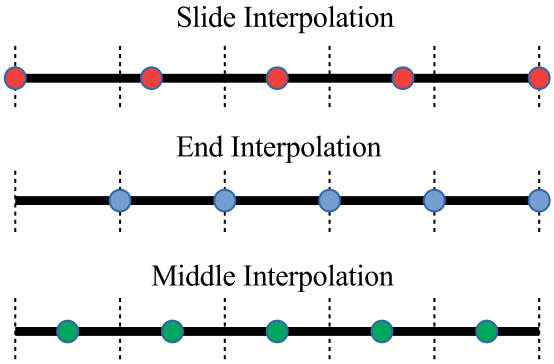


Figure 3.3: Three interpolation point positions.

3.4.2 Dynamically select area calculation parameters

The incorrect area calculation of the last valid interpolation interval is the major contributor to the average precision distortion. As shown in Figure 3.2(d), the correct calculation of the area under the PR curve segment for the last valid interpolation interval requires changing the parameters according to the actual situation. Suppose there are M valid interpolation intervals and the Mth is the last valid interpolation interval. For the interpolation intervals from 1 to M − 1, the N-point interpolation method is employed for the calculation.. For the last valid interpolation interval, the parameters of the area calculation are dynamically selected. The enhanced N-point interpolation method is derived as Equation 3.6.

$$\begin{aligned} \mathrm{AP}_{e\mathrm{N}} &= \frac{1}{\mathrm{N}} \sum_{i=1}^{\mathrm{M}-1} \mathrm{P}_{max}(\mathrm{R}_{ip}^{i}) + \mathrm{T} \times \mathrm{P}(\mathrm{R}_{ip}^{d}) \\ \mathrm{P}_{max}(\mathrm{R}_{ip}^{i}) &= \max_{\mathrm{R}':\mathrm{R}' \geq \mathrm{R}_{ip}^{i}} \mathrm{P}(\mathrm{R}') \end{aligned} \tag{3.6}$$

The number of valid interpolation intervals M and the middle interpolation points R_{ip}^{i} are calculated as follows. $\lceil * \rceil$ means rounding up.

$$\begin{aligned} \mathrm{M} &= \lceil \frac{\mathrm{N}_{tp} \times \mathrm{N}}{\mathrm{N}_{gt}} \rceil \\ \mathrm{R}_{ip}^{i} &= \frac{2i+1}{2\mathrm{N}}, i \in [0, 1, \ldots, \mathrm{N}-1] \end{aligned} \tag{3.7}$$

The valid interval width $\mathrm{T} = \mathrm{N}_{ltp}/\mathrm{N}_{gt}$. The number of true positives for the last valid interpolation interval N_{ltp} and the dynamic interpolation point for the last interpolation interval R_{ip}^{d} can be acquired as follows. $\lfloor * \rfloor$ means rounding down.

$$\begin{aligned} \mathrm{N}_{ltp} &= \mathrm{N}_{tp} - \lfloor \frac{\mathrm{N}_{gt}}{\mathrm{N}} \times (\mathrm{M}-1) \rfloor \\ \mathrm{R}_{ip}^{d} &= \frac{\lceil \frac{\mathrm{N}_{ltp}}{2} \rceil + \lfloor \frac{\mathrm{N}_{gt}}{\mathrm{N}} \times (\mathrm{M}-1) \rfloor}{\mathrm{N}_{gt}} \end{aligned} \tag{3.8}$$

For the last valid interpolation interval M, the interpolation point position R_{ip}^{d} is dynamically selected. And the interpolation interval width is changed from $1/\mathrm{N}$ to T. By the above two changes to the N-point interpolation method, the enhanced N-point interpolation method can effectively reduce average precision distortion.

Table 3.3: The average precision distortion μ caused by different interpolation point position for the last valid interpolation interval. Dynamically selects the area calculation parameters (Dyn Interp) has the best performance.

	Slide Interp	End Interp	**Dyn Interp**
11-P μ	-1.7141%	-1.7141%	**0.0036%**
40-P μ	-0.2157%	-0.2157%	**-0.0004%**

PR_{ex} is used to verify the effect of dynamically selects the area calculation parameters for the last valid interpolation interval. $\mathrm{PR}_{ex} = e^{5.4n/\mathrm{N}_{tp}-5.7} + 1$, $n = 0, 1, \ldots, \mathrm{N}_{tp}$. The number of true positive is $\mathrm{N}_{tp} = 840$. The number of ground truth is $\mathrm{N}_{gt} = 1000$. As shown in Table 3.3, the average precision distortion caused by the dynamically selected parameters is much less than other methods.

In comparing with Table 3.1, the slide interpolation used in the 11-point interpolation method and the end interpolation used in the 40-point interpolation method, the average precision distortion produced on the last valid interpolation interval is almost the same value as the average precision distortion produced cumulatively in all other interpolation intervals. It suggests that the last valid interpolation interval is a critical source of average precision distortion for the classical method.

3.4.3 Average precision calculation speed

In addition to the accuracy of average precision calculations, we are also concerned about the speed of the calculation. One of the most important advantages of the N-point interpolation method over the all-point interpolation method is that the average precision can be calculated faster. In the enhanced N-point interpolation method, the change in the interpolation point position does not lead to an increase in computational effort. The increased calculation time for dynamic selection of area calculation parameters is negligible. Therefore, the enhanced N-point interpolation method is a fast average precision calculation method as well as the classical N-point interpolation method.

3.5 Experiments and results

3.5.1 Methodology

The experiments for verifying the performance of the enhanced N-point interpolation method are conducted on the KITTI 3D object detection dataset [Geiger et al., 2012]. There are three advantages of choosing the KITTI dataset. 1) A total of 7481 labeled frames available for training and testing. It is sufficient for a comparative experiment of different average precision calculation methods. 2) The KITTI dataset contains a variety of testable tasks, including 3 detection modes, 3 classes, 3 difficulties and 2 overlap thresholds, with a total of 54 different precision-recall curves for average precision calculation. 3) KITTI is one of the most well-known autonomous driving datasets with a large number of outstanding models available for testing.

In order to compare the average precision calculation methods, three baseline detectors are adopted to generate detection proposals for average precision calculation, which are SECOND [Yan et al., 2018], PointPillars [Lang et al., 2019] and PartA2 [Shi et al., 2020a].

There are two average precision calculation methods adopted by the KITTI dataset, the 11-point interpolation method [Geiger et al., 2012] and 40-point interpolation method [Simonelli et al., 2019]. After improvement, the enhanced 11-point interpolation method and the enhanced 40-point interpolation method are obtained. In conclusion, a total of four average precision calculation methods are examined.

The target of the N-point interpolation method is to replace the all-point interpolation method with a minor cost. The more consistent the average precision calculated by the N-point interpolation method AP_N is with the all-point interpolation method AP_A, the more accurate this N-point interpolation method is. Therefore, the average precision distortion μ can be quantified by $\mu = AP_A - AP_N$.

As the range of average precision distortion varies widely, in some cases it is difficult to compare the improvement effect since the average precision distortion is almost zero. To facilitate the observation, relying on the diversity of evaluation task modes provided by the KITTI dataset, we propose to use the mean of absolute average precision distortion $|\bar{\mu}|$ metric, as shown in Equation 3.9.

$$\begin{aligned} |\bar{\mu}| &= \frac{1}{3\times 3\times 3\times 2}\sum_{m=1}^{3}\sum_{c=1}^{3}\sum_{d=1}^{3}\sum_{o=1}^{2}|\mu| \\ &= \frac{1}{54}\sum_{m=1}^{3}\sum_{c=1}^{3}\sum_{d=1}^{3}\sum_{o=1}^{2}|AP_{N|m,c,d,o} - AP_{A|m,c,d,o}| \end{aligned} \tag{3.9}$$

Where $m \in [1:2D, 2:BEV, 3:3D]$ is the task mode. $c \in [1:Car, 2:Pedestrian, 3:Cyclist]$ is the detection classes. $d \in [1:Easy, 2:Moderate, 3:Hard]$ is the difficulty of object. o is the overlap threshold for true positive. A smaller value of $|\bar{\mu}|$ indicates that this average precision calculation method is more accurate.

The standard deviation of average precision distortion σ in Equation 3.10 reflects the instability of the model performance evaluation caused by the average precision distortion.

$$\sigma = \sqrt{\frac{\sum(\mu - \bar{\mu})^2}{54}} \tag{3.10}$$

Where $\bar{\mu}$ is the mean of all average precision distortion values for the detection proposals of a baseline detector. The σ reflects the average precision distortion variation of the average precision calculation method. The smaller σ indicates that the method is more stable in evaluating the performance of the detection model.

3.5.2 Analysis of Experiment Results

The mean of absolute average precision distortion is shown in Table 3.4. It is clear that the enhanced 40-point interpolation method e40-P produces the least average precision distortion, which is usually around 0.04%. In contrast, the 11-point interpolation method 11-P and the 40-point interpolation method 40-P cause too much

average precision distortion. We consider that these two evaluation metrics may not be suitable for the KITTI dataset.

Table 3.4: Mean of absolute average precision distortion $|\bar{\mu}|$. The enhanced 40-point interpolation method e40-P has the smallest $|\bar{\mu}|$ and therefore the best accuracy of average precision calculation.

$\lvert\bar{\mu}\rvert$	11-P	40-P	e11-P	**e40-P**
SECOND	1.7393%	0.8166%	0.1733%	**0.0430%**
PointPillars	1.2262%	0.8289%	0.1177%	**0.0419%**
PartA2	1.6654%	0.5262%	0.1004%	**0.0470%**

In addition, the average precision distortion of the enhanced 11-point interpolation method e11-P is much smaller than that of the 40-point interpolation method. It indicates that the two improvements for the N-point interpolation method are successful in reducing the average precision distortion.

Compared to the 11-point interpolation method, the enhanced 11-point interpolation method reduces the average precision distortion by 90.03%, 90.40% and 93.97%, respectively. Compared to the 40-point interpolation method, the enhanced 40-point interpolation method reduces the average precision distortion by 94.73%, 94.94%, and 91.06%, respectively.

Table 3.5: Standard deviation of average precision distortion σ. The enhanced 40-point interpolation method e40-P has the smallest σ and therefore the best stability of average precision calculation.

σ	11-P	40-P	e11-P	**e40-P**
SECOND	2.2143%	0.7848%	0.1979%	**0.0578%**
PointPillars	1.5293%	0.8525%	0.1140%	**0.0541%**
PartA2	1.7240%	0.5693%	0.0945%	**0.0582%**

Table 3.5 shows the standard deviation of average precision distortion σ. Among the four methods, the most stable average precision evaluation metric is the enhanced 40-point interpolation method, and the worst is the 11-point interpolation method.

Compared to the 11-point interpolation method, the enhanced 11-point interpolation method reduces the σ by 91.06%, 92.54% and 94.51%, respectively. Compared to the 40-point interpolation method, the enhanced 40-point interpolation method reduces the σ by 92.63%, 93.65%, and 89.77%, respectively.

In conclusion, among the four N-point interpolation methods, the enhanced 40-point interpolation method is able to calculate average precision accurately and stably. For the KITTI dataset, the enhanced 40-point interpolation method can be used to reduce the average precision distortion and evaluate the performance of the detection model more reliably.

3.6 Conclusion

Average precision distortion problem seriously affects the application of the N-point interpolation method in object detection model evaluation. In order to solve the average precision distortion, we proposed an enhanced N-point interpolation method for accurate and stable average precision calculation. For this purpose, two improvements were carried out on the N-point interpolation method. 1) The interpolation point position was chosen to be the middle interpolation instead of the slide interpolation or end interpolation. 2) The area of the last valid interpolation interval was dynamically calculated, making the selected parameters more compatible with the real situation and reducing average precision distortion.

Future work will focus on the study of the relationship between the amount of data and the selection of parameter N. In combination with the enhanced N-point interpolation method, a fast and accurate average precision calculation method that could automatically adapt to different data amounts would be available.

Chapter 4

Non-Pedestrian Area Estimation

Contents

4.1 Motivation

Detecting pedestrians at high speed in high resolution (HR) images is important for preventing collisions in autonomous vehicle. The HR images provide more detail information while creating a heavy computational load, resulting in slower detection speed. The existing image optimization methods could cause more computational consumption or information loss. We try to find an image optimization model that works for the entire dataset, even for continuously collected data.

We find that the pedestrians are usually not present in the upper part of the HR image, due to the camera vertical field of view and the distance between pedestrian and camera. We named this area a non-pedestrian area (NPA). Therefore, we are able to reduce the calculation time while maintaining the detection accuracy by removing NPA from the HR image. According to this concept, we propose a novel pedestrian detection acceleration algorithm called Non-Pedestrian Area Estimation (NPAE). The NPAE algorithm estimates and removes non-pedestrian areas of the image, followed by pedestrian detection of the NPAE output image.

4.2 Introduction

The pedestrians are one of the vulnerable roles of public transportation since they tend to suffer more injuries when a collision occurs between vehicles and pedestrians [Batouli et al., 2020]. According to the World Health Organization (WHO) 2020 Road Traffic Injury Report, more than 67,500 pedestrians, cyclists and motorcyclists die in road traffic accidents every year [Calvi et al., 2020]. Injury risk factors are mainly caused by human error, such as speeding, driving under the influence of alcohol. Based on fast and accurate pedestrian detection, autonomous vehicle (AV) can avoid these kinds of human errors and significantly reduce traffic accidents and casualties.

Pedestrian detection technology has made great progress in the past decade [Brunetti et al., 2018] due to deep neural networks. As more and more layers are adopted, deep neural networks usually have higher detection accuracy. The increased complexity requires more computing resources and usually consumes additional computing time.

In an emergency, even if the braking system responds immediately, AV still needs a long braking distance. For example, the speed limit on French city roads

is 50 km/h [Goverment, 2020] and the braking distance is 14.5 m according to [Sabri and Fauza, 2018]. To ensure pedestrian safety, pedestrian detectors are required to locate targets at the farthest possible distance. Not only that, but the detector also needs to be fast enough to gain reaction time for the AV.

High-resolution (HR) image is required to detect pedestrian targets at long distances. HR image has many advantages, such as improved detection results [Bosquet et al., 2020]. It is also a basic need for autonomous driving [Huang et al., 2018]. The rich detail information provided by HR images improves the accuracy of long-distance detection. However, it increases the amount of computational consumption and time. Therefore, Pedestrian detection needs to be accelerated and kept accurate.

Most of the advanced acceleration methods are summarized in [Cheng et al., 2018], such as pruning, low-rank approximation, quantization, knowledge distillation and compact network design. All mentioned methods are based on optimizing deep neural networks for acceleration. These methods are incredible in thought and design, and have significant effects in practical.

However, we noticed that the input image is an under-optimized part of the detection process. There are two processing methods for the image input to the detector. One method is to compress the input image to a fixed size as in the [Liu et al., 2016, Redmon and Farhadi, 2017]. Another method crops part of the image area, such as [Geiger et al., 2013]. Both of these image processing methods are very convenient. Nevertheless, the compress-image method results in a loss of detection accuracy. The crop-image method can only be used with an existing dataset and acquisition environment. Changes in camera model, spatial position, and scaling require researchers to redesign the system.

We observe that the HR images obtained by autonomous vehicles contain some areas without pedestrians. We named this area a Non-Pedestrian Area (NPA). We find that there is a relationship between the region size of NPA in the image and the environmental parameters. The environmental parameters contain camera height, camera pitch angle, camera vertical field of view (FOV), the distance between the pedestrian and the camera, pedestrian height and vertical resolution of the image. Therefore, we propose the Non-Pedestrian Area Estimation (NPAE) algorithm to describe the relationship between the two factors. Afterwards, the NPAE algorithm can estimate the NPA based on the environmental parameters. Once the NPA is acquired, we remove the non-pedestrian area from the HR image and then send the

processed image to the subsequent pedestrian detector.

We first design an experiment to verify the correctness of the NPAE algorithm. Then, the algorithm acceleration performance is tested on the JAAD dataset both on the GPU platform and the CPU platform. Finally, the performance of the algorithm in extreme cases is tested on the Caltech dataset which has a plentiful variety of scenarios. Among them, the JAAD dataset contains two types of high-resolution images, 1920×1080 pixels and 1280×720 pixels. The Caltech data set contains a low-resolution 640×480 pixels image.

The NPAE algorithm is proposed for acceleration pedestrian detection. It works by estimating and cropping the Non-Pedestrian Area of the image. Due to the reduction of image size, the pedestrian detector can process faster. Moreover, the accuracy of the pedestrian detector can be maintained because the pedestrian area of the image is not modified. The NPAE algorithm has several highlights: 1) The NPAE algorithm can calculate and crop the Non-Pedestrian Area uniformly for the images acquired under the same capture conditions. 2) The NPAE algorithm does not require training and is capable of accelerating a variety of state-of-the-art pedestrian detectors. 3) The proposed algorithm is an image data optimization method that can also be used in a variety of applications such as image segmentation, object tracking.

4.3 Related Work

4.3.1 Pedestrian Detectors

Integrate Channel Features (ICF) is a classic pedestrian detection method. The ICF detector involves channel feature pyramids and boosted classifiers. Subsequent work such as Aggregated Channel Features (ACF), Locally Decorrelated Channel Features (LDCF), and SquaresChnFtrs (SCF) are based on ICF for further improvement [Zhang et al., 2016].

In recent years, machine learning methods such as SVM, ELM, ANN, CNN have gradually replaced manual design rules. In the field of pedestrian detection, deep learning has been adopted by most algorithms with excellent performances. These algorithms based on deep neural networks have higher detection accuracy, but they still need huge calculation during training and testing. The following are several classic neural network detectors:

RCNN [Girshick et al., 2014] uses the selective search to extract a large number of region proposals. These regions are converted into fixed-size images and sent to convolutional neural networks to detect and classify. Fast RCNN [Girshick, 2015] extracts features only once to generate region proposals. This task solves the problem of RCNN by repeatedly extracting features and increases the network detection speed. Faster R-CNN [Ren et al., 2015] further proposes a candidate region network (RPN), which can generate region proposals more efficiently.

YOLO [Redmon et al., 2016] uses the entire image as the input of the network and directly returns the position of the bounding box and the class it belongs to in the output layer, which greatly speeds up the detection. After that, YOLOv2 [Redmon and Farhadi, 2017] combined the advantages of RCNN-like algorithms to further improve the detection accuracy and speed.

RetinaNet [Lin et al., 2017b] is a one-stage object detector. Usually, the one-stage detector has a faster detection speed, but the detection accuracy is not as good as the two-stage detector. RetinaNet adopts the Feature Pyramid Network (FPN) from [Lin et al., 2017a] as the backbone network. The FPN fuses different levels of features through lateral connections. Due to this architecture, the network can effectively construct a rich multi-scale feature pyramid from the input image. RetinaNet also introduces a novel loss function called the focal loss to solve the foreground-background imbalance problem. These designs ensure that RetinaNet is an efficient and accurate detector. Therefore, we chose RetinaNet as the reference of the pedestrian detector to test the performance of the proposed NPAE algorithm.

4.3.2 Pedestrian Detection Acceleration

Neural Network Optimization

The current mainstream idea for neural network acceleration is to optimize the neural network structure to reduce network complexity and calculation. According to different perspectives of network optimization strategies, it can be divided into the following five categories:

Pruning [Guo et al., 2016]: The pruning method assumes that many neural network parameters are not necessary and deletes them. The pruning method preserves the structure of the neural network and reduces the computation. However, the pruning process usually is not lossless to accuracy.

Low-rank approximation [Zhang et al., 2015]: Low-rank approximation re-

duces the dimension of the convolutional kernel which will facilitate computation.

Quantization [Wu et al., 2016]: This method can be divided into two categories: (1) Scalar and vector quantization: Quantize each weight of the network into a certain element in a finite set. (2) Fixed-point quantization: Convert floating-point numbers in neural networks to fixed-point numbers.

Teacher-student network [Yim et al., 2017]: Generally, the teacher network is a large neural network or a collection of neural networks, while the student network is a compact and efficient neural network. By using the student network to fit the teacher network, the knowledge learned by the teacher network is transferred to the student network.

Compact network design [He et al., 2016]: By using methods such as 1×1 convolution and branching strategies, a more compact network structure can be designed to improve network performance.

The methods described above are excellent optimization methods. Nevertheless, these methods still have some disadvantages. A large amount of repeated training is required when applying pruning methods to neural networks. The low-rank method requires adequate design. Quantization is not suitable for all tasks and needs to convert the network, which may cause a loss of accuracy. The teacher-student method requires a lot of training and requires a good teacher network. The compact network design method requires a lot of optimization.

Image Data Optimization

There are two methods for accelerating detection by optimizing the input image: one is to compress the image, and the other is to crop the image.

Image Compression. In order to detect faster and reduce resource consumption, most methods usually compress the image size. For example, there are two similar network architectures in SSD, SSD512 and SSD300. The SSD512 network uses images with a resolution of 512×512 pixels and it has high detection accuracy. However, the detection speed is slow due to the high resolution of the images used. In order to accelerate the detection speed, [Liu et al., 2016] proposes the SSD300 structure. SSD300 uses an image with a resolution of 300×300 pixels. From the results, the SSD512 has a detection accuracy of 76.8% mean average precision (mAP) and a speed of 22 frames per second (fps). To increase detection speed, the detection accuracy of SSD300 has been reduced by 2.5%.

The image compression methods increase the speed of detection, but it also

reduces the accuracy of detection. This is because compressing the image causes some of the target information to be lost, which interferes with detection.

Image cropping. Image cropping is also a very widely used method. The usual method is to crop the region of interest (RoI) in the image. For example, the image area affected by pincushion distortion effect is cropped in the KITTI dataset [Geiger et al., 2013]. The image resolution is cropped from 1392×512 pixels to 1242×375 pixels, a reduction of 34.65% pixels. Reducing the number of pixels will reduce the amount of computation and therefore accelerate the detection process.

The advantage of the image cropping method is that it is quick and easy to understand. The disadvantage is that when there is less cropping, the acceleration is less effective. When there is too much cropping, the target will be cropped. Therefore, it is necessary to get all the data first, count the target distribution and then manually adjust the cropping area. But it is difficult to meet this condition in practice

With different design concept, neural network optimization and image data optimization are both designed to accelerate pedestrian detection. The neural network optimization methods are concerned with improving accuracy and speed by finding a better detector structure. The image data optimization methods allow neural network to process less data by reducing the image size. The image data optimization methods are used in combination with optimized neural networks to take advantage of both acceleration methods for pedestrian detection. In [Redmon et al., 2016], a neural network named YOLO is proposed. The YOLO neural network first compresses the size of the input images to 224×224 pixels. Then, high-speed detection is achieved through a novel one-stage detector design. [Liu et al., 2016] introduces two architectures called SSD300 and SSD512, which compress the input images to 300×300 pixels and 512×512 pixels respectively. This design attempts to make a trade-off between accuracy and speed of detection.

However, the existing image data optimization methods lack accurate image processing. Thus, we propose the Non-Pedestrian Area Estimation (NPAE) algorithm. The NPAE algorithm is a method for image data optimization to accelerate pedestrian detection. Different from the classical image data optimization methods, the NPAE algorithm estimates the Non-Pedestrian Area (NPA) in the image based on the parameters of the image capture environment. The advantage of this processing is that the image size is reduced, while the areas containing pedestrian are preserved. Therefore, the NPAE algorithm could accelerate pedestrian detec-

tion and maintain detection accuracy. Compared to neural network optimization methods, the NPAE algorithm has several advantages: 1) The proposed algorithm does not require any training process. 2) The optimization process does not affect the detection accuracy. 3) The NPAE algorithm could be used in the neural network optimization methods together to accelerate pedestrian detection.

4.4 Algorithm Design

4.4.1 Pedestrian Location Analysis

To present the basic idea of this paper, we selected two datasets for observation and experimentation.

JAAD dataset: The dataset includes high-resolution (HR) images in two formats, 1920 × 1080 pixels and 1280 × 720 pixels. It is a representative dataset of HR images for pedestrian detection. The two resolutions allow us to observe the distribution of pedestrians in different situations.

Caltech dataset: The dataset has the images with low resolution (LR) of 640 × 480 pixels. The Caltech dataset does not contain HR images. However, it contains a rich set of scenes and is widely adopted by existing methods. Thus this LR image dataset is used for comparison experiments. Both the JAAD dataset and the Caltech dataset have a frame rate of 30 fps.

For the selected dataset, we will count the distribution of pedestrians. Therefore, it is necessary to select one of the pedestrian's attribute parameters as the statistical object. We define that the pedestrian bounding box is located by two points, P1 and P2, as shown in Figure 4.1. Where P1 = $(P1_x, P1_y)$ indicates the top-left vertex, P2 = $(P2_x, P2_y)$ indicates the right bottom vertex.

Since pedestrians will appear at random locations, $P1_x \in [0, X_{max}]$, $P2_x \in [0, X_{max}]$. The range of $P1_y$ and $P2_y$ cannot be determined. We categorized the images into three groups by resolution: 1920 × 1080, 1280 × 720 and 640 × 480. All P1 and P2 points are plotted in the image space of the corresponding resolution, respectively, as shown in Figure 4.2.

The purpose of analyzing the three sets of points P1 and P2 is to determine whether the Non-Pedestrian Area (NPA) is widely present in the image.

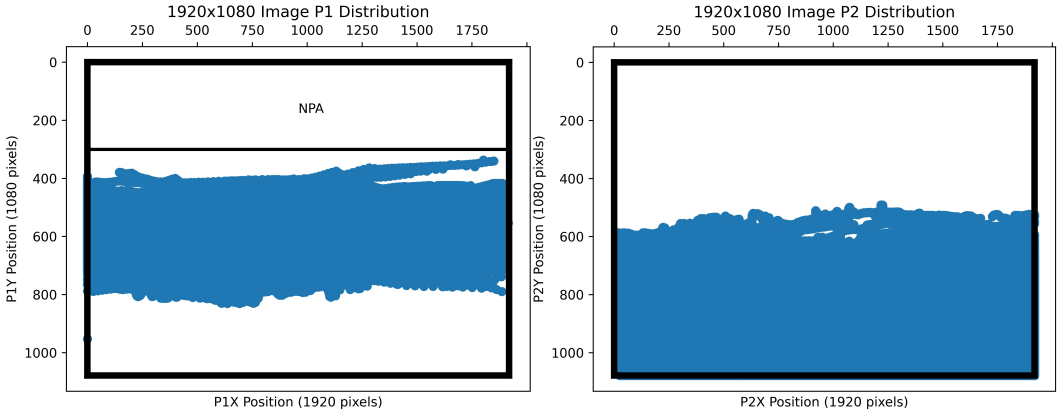
P1 represents the top boundary of the pedestrian bounding box, so we need to find NPA_{top} in the upper half of the image. NPA_{top} can be found in the three sets of

Figure 4.1: Definition of pedestrian bounding box (bbox). Bbox is jointly determined by point P1 and point P2.

images with different resolutions in Figure 4.2(a), 4.2(c) and 4.2(e). In the HR image, the NPA$_{top}$ region is larger. In LR images, the region of NPA$_{top}$ is smaller. There are some exceptions to the NPA$_{top}$ region in LR images, which we will discuss in subsection 4.4.3.

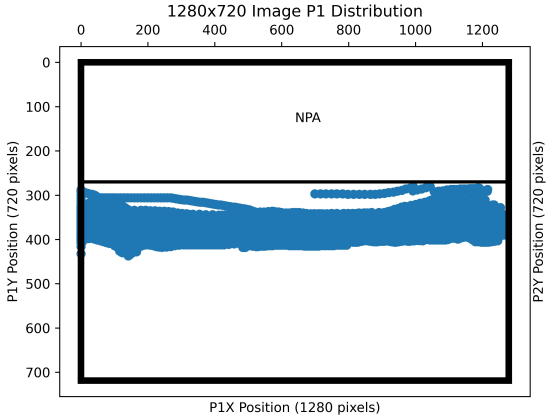
P2 represents the bottom boundary of the pedestrian bounding box, so we need to find NPA$_{bottom}$ in the lower half of the image. NPA$_{bottom}$ does not exist in the three sets of images with different resolutions in Figure 4.2(b), 4.2(d) and 4.2(f).

This situation is not a coincidence. For a better view, the camera is usually set up on top of the autonomous vehicle. Hence $h_c = h_v$, where h_c is the camera height and h_v is the vehicle height. In general, vehicle height h_v and pedestrian height h_p have $h_v > h_p/2$. Then $h_c > h_p/2$, which means that the area where pedestrian appears in the image is more towards the bottom. This is the reason why NPA only exists at the top of the image. In this paper, we use NPA to refer to NPA$_{top}$. From the analysis of the dataset, we can see that NPA is present. The next step is to analyze the environmental parameters related to the NPA and find an algorithm to estimate NPA.

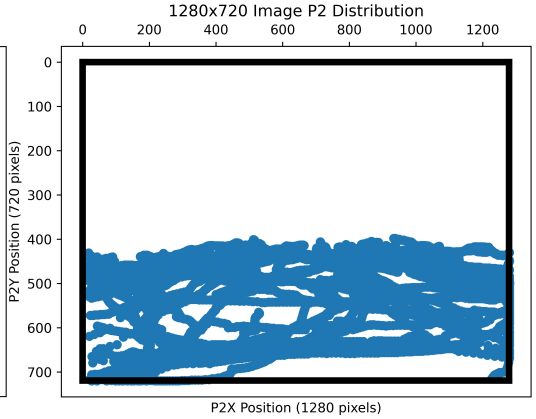


(a) JAAD 1920 × 1080 P1.

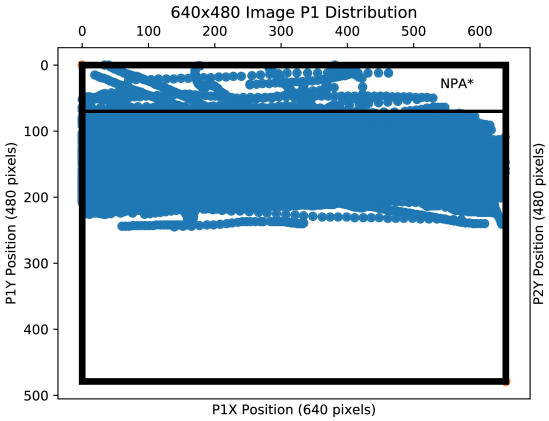
(b) JAAD 1920 × 1080 P2.



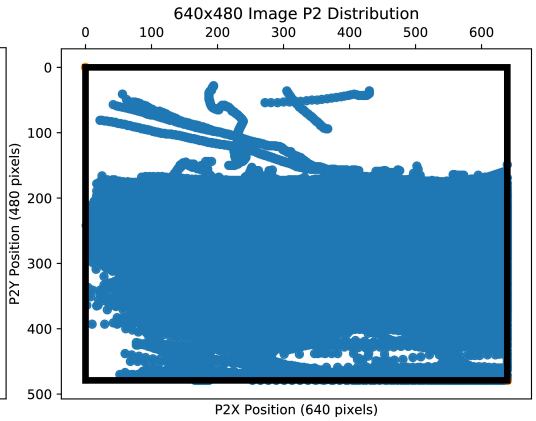
(c) JAAD 1280 × 720 P1.



(d) JAAD 1280 × 720 P2.



(e) Caltech 640 × 480 P1*.



(f) Caltech 640 × 480 P2.

Figure 4.2: P1 is the upper left corner of the pedestrian bounding box. P2 is the lower right corner of the pedestrian bounding box. Non-pedestrian areas are present in the statistics at point P1 in all three datasets. NPA is not present in the statistics at point P2. Thus for the dataset, the non-pedestrian areas are predominantly present at the top of the image. There are special scenarios in the Caltech dataset that need to be analyzed, so non-pedestrian areas are marked as NPA*.

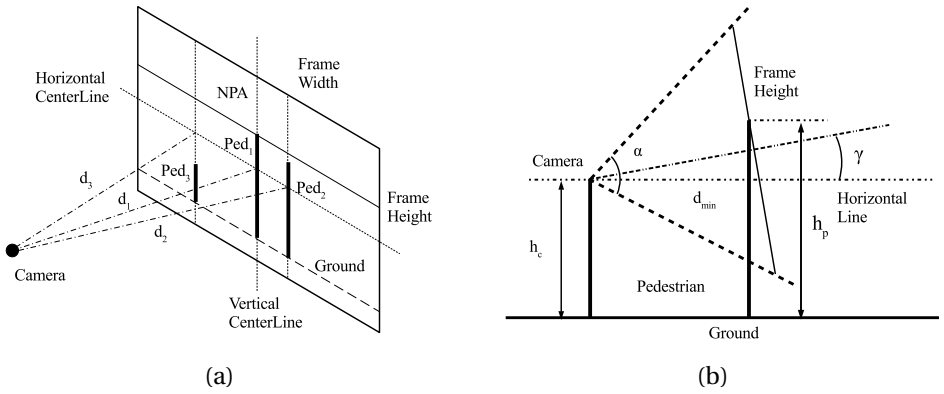


Figure 4.3: Analyze and build NPAE algorithms for single images. (a) Analysis of non-pedestrian area at the top of the image containing multiple pedestrians. (b) Analysis of the environmental parameters involved in NPAE algorithm.

4.4.2 Non-Pedestrian Area Estimation for Single Image

Our goal is to find suitable non-pedestrian areas for all images in the dataset. The situation in the dataset is more complicated. It contains a large number of pedestrians, with different camera postures and different collection equipment. Therefore, we first try to find the optimal NPA for a single image. Then we will discuss how to find a common applicable NPA on the dataset.

To accurately estimate the non-pedestrian area in a single image, it is necessary to analyze the highest position of the pedestrian bounding box P1 point in the image. As shown in Figure 4.3(a), there are three pedestrians, Ped_1, Ped_2 and Ped_3. The distance between camera and pedestrian is d_1, d_2 and d_3, respectively. Ped_1 is on the vertical centre line of the image, d_1 is perpendicular to the plane of the image. Therefore $d_1 > d_2, d_3$. According to the imaging rule of the camera, the closer the pedestrian is, the larger the image is. Suppose that the two pedestrians of Ped_1 and Ped_2 are the same height. Since Ped_1 is closer to the camera, the P1 point of its bounding box is higher than Ped_2's.

Figure 4.3(b) establishes a model of pedestrian and camera. Assume that the pedestrian is located in the center of the image. At this position, the pedestrian's imaging area is the largest and has the highest P1 point. Several environmental parameters are used: h_c is the camera height, γ is the camera pitch angle, α is the camera vertical field of view (FOV), d_{pc} is the distance between the pedestrian and the camera, h_p is the pedestrian height, px_v is the vertical resolution of the image, NPA is the Y-axis coordinate of the pedestrian bounding box P1 point according to

the environmental parameters.

We first consider the scenario where the pitch angle γ of a camera is 0. The goal of the NPAE algorithm is to calculate the vertical coordinate of a point P on the image. The height of the camera is h_c, the height of point P is h_p, and the distance of point P from the camera is d. According to the camera vertical FOV α and d, the observable length $l_d = 2d\tan(\alpha/2)$ can be calculated. Assume that the vertical coordinate of point P in the image is Y, and the vertical resolution of the image is px_v. According to the proportional relationship we can get: $Y : px_v = (l_d/2 - (h_p - h_c)) : (l_d/2)$. Thus $Y = px_v/2(1 - (h_p - h_c)/(d\tan(\alpha/2)))$. Here the Y represents the Y-axis coordinate of the non-pedestrian area. According to the above analysis, the NPAE algorithm equation with γ is shown in Equation (4.1):

$$\mathrm{NPA} = \frac{px_v}{2}\{1 - \frac{(h_p - h_c)\cos\gamma - d_{pc}\sin\gamma}{[d_{pc}\cos\gamma + (h_p - h_c)\sin\gamma]\tan\frac{\alpha}{2}}\} \tag{4.1}$$

In order to verify NPAE algorithm in Equation 4.1, six environmental parameters are needed. After recording and inputting the required parameters of Equation 4.1, the NPAE algorithm can be verified by comparing the calculated value of NPA with the actual value. Since the dataset does not contain the required environmental parameter, a validation experimental platform is built.

The parameters in the validation experimental platform are: The height of the pedestrian h_p is 1.90 m. The camera height h_c is 1.17 m. The vertical resolution of the image px_v is 1080 pixels. The camera vertical FOV α is 52 degrees. The camera pitch angle γ is set at 0.5 degrees, 3.5 degrees, and 10 degrees in sequence. The distance between the pedestrian and the camera d_{pc} is set at 2.0 m, 2.5 m, 3.0 m, 3.5 m, and 4.0 m in sequence. The vertical coordinate of the P1 point of the pedestrian bounding box $\mathrm{P1}_y$ is manually measured. The experimental results are shown in Table 4.1.

It can be seen from Table 4.1 that the NPA value estimated by the NPAE algorithm is consistent with the $\mathrm{P1}_y$. The *Dissimilarity* index are adopted to measure the similarity between NPA and $\mathrm{P1}_y$. There are a lot of similarity calculation methods, such as Euclidean metric, Manhattan distance and Chebyshev distance. The outputs of these methods are the distance between the two sets of data, which does not visually show the degree of similarity. While the output of cosine similarity shows the similarity in the form of percentage. This makes it easier to determine the degree of similarity between NPA and $\mathrm{P1}_y$. Thus, we choose to use cosine similarity

Table 4.1: Verify the correctness of the NPAE algorithm. The dissimilarity is calculated by $P1_y$ and NPA. A smaller value means that the two vectors are more similar. The six environmental parameters used: camera height h_c, camera pitch angle γ, camera vertical view angle α, pedestrian-camera distance d_{pc}, pedestrian height h_p, vertical resolution px_v. $P1_y$ is the P1 point Y-axis value of ground truth. NPA is the non-pedestrian area calculated by the NPAE algorithm. px means pixel and dg means degree.

px_v (px)	h_p (m)	h_c (m)	α (dg)	γ (dg)	d_{pc} (m)	**$P1_y$ (px)**	**NPA (px)**	Dissimilarity
1080	1.90	1.17	52	0.5	2.0	**145**	**147**	
1080	1.90	1.17	52	0.5	2.5	**223**	**227**	
1080	1.90	1.17	52	0.5	3.0	**277**	**281**	4.53×10^{-5}
1080	1.90	1.17	52	0.5	3.5	**315**	**319**	
1080	1.90	1.17	52	0.5	4.0	**350**	**348**	
1080	1.90	1.17	52	3.5	2.0	**216**	**211**	
1080	1.90	1.17	52	3.5	2.5	**296**	**289**	
1080	1.90	1.17	52	3.5	3.0	**349**	**341**	2.74×10^{-6}
1080	1.90	1.17	52	3.5	3.5	**386**	**379**	
1080	1.90	1.17	52	3.5	4.0	**415**	**407**	
1080	1.90	1.17	52	10.0	2.0	**337**	**344**	
1080	1.90	1.17	52	10.0	2.5	**416**	**418**	
1080	1.90	1.17	52	10.0	3.0	**468**	**469**	3.37×10^{-5}
1080	1.90	1.17	52	10.0	3.5	**509**	**506**	
1080	1.90	1.17	52	10.0	4.0	**536**	**533**	

to measure the degree of similarity between the two vectors. The formula of cosine similarity is:

$$sim(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{||\boldsymbol{x}||||\boldsymbol{y}||} \tag{4.2}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are:

$$\begin{aligned} \boldsymbol{x} &= (\mathrm{P1}_{y1}, \mathrm{P1}_{y2}, ..., \mathrm{P1}_{y5}) \\ \boldsymbol{y} &= (\mathrm{NPA}_1, \mathrm{NPA}_2, ..., \mathrm{NPA}_5) \end{aligned} \tag{4.3}$$

and $||\cdot||$ is the Euclidean norm of vector:

$$\begin{aligned} ||\boldsymbol{x}|| &= \sqrt{\mathrm{P1}_{y1}^2 + \mathrm{P1}_{y2}^2 + \cdots + \mathrm{P1}_{y5}^2} \\ ||\boldsymbol{y}|| &= \sqrt{\mathrm{NPA}_1^2 + \mathrm{NPA}_2^2 + \cdots + \mathrm{NPA}_5^2} \end{aligned} \tag{4.4}$$

The measure computes the cosine of the angle between vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. The closer the cosine value to 1, the smaller the angle and the greater the match between vectors [Han et al., 2011]. We can then get the dissimilarity by:

$$dissim = 1 - sim(\boldsymbol{x}, \boldsymbol{y}) \tag{4.5}$$

Since the results of dissimilarity are very close to 0, that means the proposed Non-Pedestrian Area Estimation algorithm accurately calculates the non-pedestrian area.

4.4.3 Non-Pedestrian Area Estimation for Multiple Images

NPAE algorithm is designed to find the non-pedestrian area for the dataset. There is a difference between applying the NPAE algorithm to a dataset and applying the algorithm to a single image. When using the NPAE algorithm on a dataset, all six environment parameters need to be set to constants. In this way, the NPAE algorithm can uniformly remove non-pedestrian areas for the images in the dataset based on the set environmental parameters. However, the situation is further complicated by the large number of images contained in the dataset. The six environmental parameters used by NPAE need to be revisited.

Camera vertical FOV α: The camera vertical FOV does not change for a dataset or an automated vehicle once the camera has been placed. Camera vertical FOV is constant.

Vertical resolution px_v: Similar to the camera vertical FOV, the vertical resolution does not change after the camera is setup. Vertical resolution is constant.

Camera height h_c: As the vehicle runs, the camera height changes slightly under varying slope and vibration conditions. Since this change is instantaneous. Eventually it will return to its initial state. Therefore, camera height can be considered as a constant.

Camera pitch angle γ: The camera pitch angle will slightly change in the car movement. Similar to the camera height, camera pitch angle can be also considered as a constant.

Pedestrian height h_p: The pedestrian height in the dataset varies and is not available. According to subsection 4.4.1, the highest pedestrian height determines the value of NPA when the pedestrians are at the same location. Therefore we refer to [Visscher, 2008] and set h_p using the largest value of the average country citizen height in the world, thus h_p = 1.8 m.

Pedestrian-camera distance d_{pc}: According to subsection 4.4.1, the smaller pedestrian-camera distance for pedestrians with the same height determines the value of NPA. Although pedestrians may appear at any location, the d_{pc} will usually not be 0 due to the vehicle width. We refer to the average size of cars and set pedestrian-camera distance d_{pc} = 1.2 m.

The camera vertical FOV, vertical resolution, camera height, and camera pitch angle obtained from the dataset along with the predefined pedestrian height and pedestrian-camera distance are the environmental parameters required for the NPAE algorithm. The NPA value for the dataset can be obtained by entering these parameters into Equation 4.1.

Taking into account the following special conditions that may encounter: 1) The pedestrian height is higher than the assumed value. 2) The pedestrian distance from the camera is less than the assumed value. 3) The pedestrian is not in the same plane as the camera. These special pedestrian targets need to be carefully investigated. A discussion of special conditions follows.

The NPA is used to remove the horizontal pixels lines from 0 to the value of NPA. NPAE not only optimizes the image, but also changes the coordinates of the bounding boxes P1 and P2. The modified $P1_{new}$ and $P2_{new}$ can be calculated from Equa-

(a) (b)

Figure 4.4: Two special pedestrian targets in red bounding boxes. (a) The pedestrian is not on the ground. (b) The camera angle of view is small and the distance between the pedestrian and the camera is short.

tion 4.6.

$$\begin{aligned} \mathrm{P1}_{new} &\begin{cases} \mathrm{P1}_{x\_new} = \mathrm{P1}_x \\ \mathrm{P1}_{y\_new} = \mathrm{P1}_y - \mathrm{NPA} \end{cases} \\ \mathrm{P2}_{new} &\begin{cases} \mathrm{P2}_{x\_new} = \mathrm{P2}_x \\ \mathrm{P2}_{y\_new} = \mathrm{P2}_y - \mathrm{NPA} \end{cases} \end{aligned} \tag{4.6}$$

The X-axis coordinates of P1 and P2 have not changed, only the Y-axis coordinates have changed. There are three cases of $\mathrm{P1}_{y\_new}$ and $\mathrm{P2}_{y\_new}$:

I : $\mathrm{P1}_{y\_new} > 0$ and $\mathrm{P2}_{y\_new} > 0$.

In this case, the pedestrians in the processed image are complete. The processing does not have any effect on the pedestrian detection results. This is the most common state.

II : $\mathrm{P1}_{y\_new} < 0$ and $\mathrm{P2}_{y\_new} < 0$.

In this case, some pedestrians in the original images will be removed. This type of pedestrian is usually located above certain objects. There is no way that the autonomous vehicle can hurt this type of pedestrian. We found this type of pedestrian in the Caltech dataset, as shown in Figure 4.4(a). Pedestrians on the overpass are not the targets we are interested in. Dealing with such pedestrian target is a waste of processing resources. Therefore, eliminating such pedestrian targets will not affect the pedestrian detection accuracy.

III : $\mathrm{P1}_{y\_new} < 0$ and $\mathrm{P2}_{y\_new} > 0$.

In this case, some part of the pedestrian will be removed. To cause this, pedestrians have to be close enough to the camera and the camera's angle of view should be small enough. We only found some samples in the Caltech dataset, as shown in Figure 4.4(b). The area above the pedestrian's head in the red box will be partially removed, but the remaining pedestrian image still occupies a large area in the image. Due to problems such as object occlusion and pedestrian overlap, the pedestrian detector should have the ability to detect part of the pedestrian. Missing some part of the pedestrian image will not cause the detection result to fail.

Based on the above analysis, we conclude that the NPAE algorithm does not have a negative impact on the pedestrian detection results.

4.5 Experiments and Results

4.5.1 Experiment preparation

Datasets

Two pedestrian datasets are adopted, JAAD dataset [Rasouli et al., 2017] and Caltech dataset [Dollár et al., 2009].

JAAD: The JAAD dataset is developed by York University. It has various weather conditions such as sunny, cloud, rain and snow. It is collected from three cities with complex traffic scenarios and a large number of pedestrians. 2786 pedestrians are labelled in a total of 389545 bounding boxes. JAAD dataset has 336 videos with a resolution of 1920 × 1080 pixels and 10 videos with a resolution of 1280 × 720 pixels. The frame rate is 30 fps, which leads to a total of 82032 frames. There are 79812 frames with the resolution of 1920 × 1080 pixels. 63850 frames are used for training and 15962 frames are used for testing. There are 2220 frames with the resolution of 1280 × 720 pixels. 1776 frames are used for training and 444 frames are used for testing.

Caltech: Caltech dataset is developed by the California Institute of Technology. The dataset contains a large number of image frames, challenging low resolution (LR) images and images of occluded targets. The dataset consists of 10 hours of video with 2300 pedestrians marked. There are over 350000 bounding boxes. The resolution of the image is 640 × 480 pixels and the frame rate is 30 fps. 146622 frames in total are adopted in the experiment. 122185 frames are used for training and 24437 frames are used for testing.

Although our goal is to accelerate high resolution image pedestrian detection, experiments on low resolution images are still valuable. The Caltech dataset has an equal number of pedestrian bounding boxes as the JAAD dataset. However, the situation in the Caltech dataset is more complex due to the different environmental parameters set. These complex environmental conditions could test the robustness of the proposed algorithm.

Parameters for NPAE

There are six environmental parameters: camera height h_c, camera pitch angle γ, camera vertical FOV α, pedes- trian-camera distance d_{pc}, pedestrian height h_p, vertical resolution px_v.

According to [Rasouli et al., 2017], for JAAD with a resolution of 1920×1080: Vertical resolution $px_v = 1080$ pixels. Camera vertical FOV $\alpha = 73°$. Camera pitch angle $\gamma = 0°$. Camera height $h_c = 1.6$ m. For JAAD with a resolution of 1280×720: Vertical resolution $px_v = 720$ pixels. Camera vertical FOV $\alpha = 73°$. Camera pitch angle $\gamma = 0°$. Camera height $h_c = 1.6$ m. For Caltech with a resolution of 640×480: Vertical resolution $px_v = 480$ pixels. Camera vertical FOV $\alpha = 27°$. Camera pitch angle $\gamma = 0°$. Camera height $h_c = 1.6$ m. Based on the conclusions of the analysis in subsection 4.4.3, we set pedestrian height $h_p = 1.8$ m, pedestrian-camera distance $d_{pc} = 1.2$ m.

Considering the car shudder, the Camera pitch angle γ is varied between $[-1°, 1°]$. The adjusted environmental parameters can be used to derive the corresponding NPA for each dataset according to Equation 4.1. The NPA interval for JAAD 1920×1080 is between $\text{NPA}_{\gamma=-1}$ and $\text{NPA}_{\gamma=1}$. We take the minimum value, so NPA = 405 pixels. To facilitate the experiment, the value of NPA is modified using Equation 4.7.

$$\text{NPA}_{mod} = \text{NPA} - (\text{NPA} \mod 10) \tag{4.7}$$

Equation 4.7 is not obligatory. The purpose of using this operation is simply to make the experimental comparison clearer. The effect of this operation on detection performance is very slight. Using the same processing method, the NPA value of JAAD 1280×720 is 270 pixels. For the Caltech low resolution dataset, we want to verify the proposed NPAE algorithm under extreme situations. The γ is not processed. We still applied Equation 4.7. The NPA value of Caltech 640×480 is 70 pixels.

Parameters for Pedestrian Detector

The RetinaNet is adopted to detect pedestrians. There are several versions of RetinaNet, depending on the backbone used. We chose the resnet50 [He et al., 2016] as the backbone network. This network offers a better balance between detection performance and speed.

4.5.2 Experiments

Experiments are performed separately on high resolution (HR) images and low resolution (LR) images. The purpose of the experiments on HR images is to measure the performance of the NPAE algorithm acceleration. The experiments on LR images focus on the robustness of the NPAE algorithm in complex situations.

To compare the effects of detection acceleration, we designed comparison experiments for each resolution image. In general, the experiments are compared with two other image data optimization acceleration methods. The image compression method compresses the image to the same resolution as the NPAE-processed image. Since the NPAE algorithm is essentially an image cropping method, we compare the results of acceleration with different degrees of cropping. We also count the number of pedestrian targets affected by the image cropping method. There are two categories, targets that are partially cropped (PR) and targets that are completely removed (CR).

The evaluation metric used in this paper is the Average Precision (AP) [Zhu, 2004]. It is a typical object detection task evaluation metric. We record the runtime of pedestrian detection accelerated by different image data optimization methods. The CPU experimental platform is Intel Xeon W-2104 with 16GB RAM. The GPU experimental platform is Nvidia Quadro P5000. Most of the current pedestrian detection algorithms are based on deep learning. The common platforms for deep learning frameworks are CPU and GPU, therefore our experiments focus on these two platforms. The experimental results on the high resolution images are shown in Table 4.2 and Table 4.3.

We train RetinaNet using three resolution images obtained from two datasets. All images used for training are not cropped by the NPAE algorithm. We use the random rotation and random cropping to augment the input images. RetinaNet is trained for a total of 50 epochs with a learning rate of 1e-5.

After training, we combine the NPAE algorithm with RetinaNet for testing. Dur-

Table 4.2: The experimental results on JAAD 1920 × 1080 pixels test data. In combination with the proposed NPAE algorithm (*NPAE* in the table) the detection speed on the GPU platform is 8.36 fps improved by 46.92%. The detection speed on the CPU platform is 0.35 fps, which is a 52.17% improvement. *AP* mean average precision. *FPS* mean frame per second on gpu platform. *RT* mean runtime. Ped_{PC} mean partially cropped pedestrian. Ped_{CR} mean completely removed pedestrian. *G* mean GPU platform. *C* mean CPU platform.

Model	Resolution (pixels)	*AP* (%)	$\mathrm{FPS_G}$	$\mathrm{RT_G}$ (ms)	$\mathrm{FPS_C}$	$\mathrm{RT_C}$ (ms)	Ped_{PC}	Ped_{CR}
RetinaNet	1920 × 1080	84.60	5.69	175.7	0.23	4307.8	0	0
Compress	1920 × 680	61.75	8.36	119.6	0.35	2819.9	0	0
Crop-150	1920 × 930	82.94	6.04	165.4	0.26	3769.1	0	0
Crop-200	1920 × 880	87.78	6.38	156.7	0.27	3584.9	0	0
Crop-250	1920 × 830	82.59	7.12	140.4	0.29	3381.8	0	0
Crop-300	1920 × 780	86.74	7.58	131.8	0.31	3191.8	0	0
Crop-350	1920 × 730	83.01	7.91	126.4	0.33	2994.3	2	0
NPAE(Crop-400)	1920 × 680	**84.85**	**8.36**	**119.5**	**0.35**	**2813.2**	**33**	**0**
Crop-450	1920 × 630	84.03	9.14	109.4	0.38	2625.2	348	0
Crop-500	1920 × 580	81.50	9.50	105.2	0.41	2421.7	2657	3

Table 4.3: The experimental results on JAAD 1280 × 720 pixels test data. In combination with the proposed NPAE algorithm (*NPAE* in the table) the detection speed on the GPU platform is 14.42 fps improved by 39.19%. The detection speed on the CPU platform is 0.82 fps, which is a 60.78% improvement. *AP* mean average precision. *FPS* mean frame per second on gpu platform. *RT* mean runtime. Ped_{PC} mean partially cropped pedestrian. Ped_{CR} mean completely removed pedestrian. *G* mean GPU platform. *C* mean CPU platform.

Model	Resolution (pixels)	*AP* (%)	FPS_G	RT_G (ms)	FPS_C	RT_C (ms)	Ped_{PC}	Ped_{CR}
RetinaNet	1280 × 720	60.35	10.23	97.7	0.51	1926.2	0	0
Compress	1280 × 450	22.97	14.40	69.4	0.83	1196.8	0	0
Crop-150	1280 × 570	59.61	11.96	83.6	0.63	1570.6	0	0
Crop-200	1280 × 520	61.23	12.69	78.8	0.70	1424.6	0	0
Crop-250	1280 × 470	59.25	14.06	71.1	0.76	1304.6	0	0
NPAE(Crop-270)	1280 × 450	**60.91**	**14.24**	**70.2**	**0.82**	**1206.5**	**0**	**0**
Crop-300	1280 × 420	58.81	14.90	67.1	0.84	1177.5	20	0
Crop-350	1280 × 370	51.71	16.07	62.2	0.98	1018.7	252	0

Table 4.4: The experimental results on Caltech 640 × 480 pixels test data. There is limited acceleration on low resolution images. There are pedestrian targets that are partially cropped after processing by the NPAE algorithm, but the detection accuracy is not much reduced. *AP* mean average precision. *FPS* mean frame per second on gpu platform. *RT* mean runtime. Ped_{PC} mean partially cropped pedestrian. Ped_{CR} mean completely removed pedestrian. *G* mean GPU platform. *C* mean CPU platform.

Model	Resolution (pixels)	AP (%)	FPS_G	RT_G (ms)	FPS_C	RT_C (ms)	Ped_{PC}	Ped_{CR}
RetinaNet	640 × 480	66.40	22.52	44.4	1.48	674.5	0	0
Compress	640 × 410	49.23	23.98	41.7	1.70	587.3	0	0
Crop-40	640 × 440	53.01	22.98	43.5	1.60	623.6	41	5
Crop-60	640 × 420	58.67	23.64	42.3	1.66	602.4	68	17
NPAE(Crop-70)	640 × 410	**58.20**	**23.81**	**42.0**	**1.70**	**586.5**	**90**	**27**
Crop80	640 × 400	43.11	23.42	41.7	1.75	568.7	177	36

ing the test, the proposed algorithm accelerates pedestrian detection. The algorithm flow of using the NPAE algorithm to accelerate pedestrian detection is shown in Algorithm 4.1.

Algorithm 4.1 The procedure of NPAE algorithm accelerates pedestrian detection.

Input: *Environmental parameters*: Obtained from the input image collection environment.

Output: Pedestrian bounding boxes.

1: Calculate NPA value: Input *environmental parameters* to the NPAE algorithm.
2: Crop images: Crop all *images* according to the NPA value. The original image size is (H,W), and the cropped image size is (H-NPA, W).
3: Calculate pedestrian bounding box: The cropped images are sent to the pedestrian detector to get the pedestrian bounding boxes.

In Table 4.2, RetinaNet is the baseline. The detection result has 84.80% average precision (AP). It takes RT_G = 175.7 milliseconds to process an image on the GPU, which means 5.69 frames per second (FPS_G). On the CPU, the time to detect an image RT_C is 4307.8 milliseconds, which means 0.23 frames per second (FPS_C). The proposed NPAE algorithm reduces the detection time of RetinaNet to 119.5 ms and 2813.2 ms on two platforms. The detection speed is 8.36 fps on GPU and 0.35 fps on CPU. The detection speed is increased by 46.92% and 52.17% respectively. After NPAE acceleration, the AP of the detector RetinaNet remains at 84.85%.

Image compression method, as C$ompress$ mentioned in Table 4.2, can also accelerate detection. However, there is a significant loss of accuracy. The experimental detection accuracy AP is only 61.75%. The results can be further improved to a certain extent by extensive multi-scale training. Nevertheless, this improvement is limited and leads to a more complex process.

NPAE is an accurate image cropping method. We have also labeled it *Crop-NPA*. In order to compare the performance of NPAE, seven experiments with different cropping values are designed in Table 4.2. These experiments are named *Crop-Value* and *Value* is the number of rows of cropped pixels. In addition to analyzing the detection accuracy *AP* and detection speed *FPS*/*RT*, the pedestrian targets affected by the cropping should also be considered. The partially cropped pedestrian target number is recorded as Ped_{PC}, and the completely removed pedestrian is labeled as Ped_{CR}.

In terms of acceleration, the speed increases as the degree of shearing increases. In terms of detection accuracy, it first remains constant and then decreases. The in-

flection point occurs around the NPA value estimated by the NPAE algorithm. The Ped_{PC} and Ped_{CR} values also show an inflection point around the NPA value. This pattern means that more and more pedestrian area information is lost as the cropping range exceeds the NPA value. This results in a decrease in detection accuracy.

The same conclusion can be drawn from the experimental results on the JAAD 1280×720 dataset in Table 4.3. On the GPU platform, the proposed NPAE algorithm increases the detection speed by 39.19% from 10.23 fps to 14.24 fps relative to baseline. On the CPU platform, detection is 60.78% faster, from 0.51 fps to 0.82 fps. The NPAE algorithm increases the speed significantly while the detection accuracy remains unaffected.

The experimental results on the low resolution images are shown in Table 4.4. There is little room for image data optimization on low resolution images. Nevertheless, the similar pattern as in Table 4.2 and Table 4.3 can also be seen. Despite the decrease in accuracy, the speed has increased. What is enlightening is the quantity of pedestrian partially cropped Ped_{PC} and pedestrian completely removed Ped_{CR}.

The detection accuracy of the accelerated detectors remains consistent from the *Crop-40* method to the NPAE algorithm. Although some pedestrians are partially cropped, it does not significantly affect the detection results. This means that the detector can still get the information it needs from Ped_{PC}. With further cropping, such as *Crop-80*, the detection accuracy decreases dramatically. At this point, the detector no longer has enough information to ensure the correctness of the detection results. There are 27 pedestrians that are completely removed indicated by Ped_{CR}. These targets are consistent with the situation discussed in subsection 4.4.3.

From the experimental results, the proposed NPAE algorithm is suitable for optimizing high resolution images. The acceleration obtained for pedestrian detection is significant and the algorithm does not affect the detection accuracy. The NPAE algorithm can also be applied on low resolution images, but with ordinary results.

4.6 Conclusion

In this chapter, we presented an algorithm for accelerating pedestrian detection with high resolution images, named Non-Pedestrian Area Estimation (NPAE). Based on six environmental parameters, the NPAE algorithm can accurately estimate the non-pedestrian areas in the dataset. Therefore, the proposed algorithm cropped the image more accurately compared to other image data optimization methods.

Because the image area was reduced and the object information was preserved, the pedestrian detector gained acceleration while maintaining accuracy.

Two high resolution images, 1920×1080 and 1280×720, and two different computing platforms, GPU and CPU, were used for the experiments. In the GPU platform, the detection speed was increased by 46.92% and 39.19%, respectively. In the CPU platform, the detection speed was increased by 52.17% and 60.78%, respectively. All experiments have shown that the proposed algorithm does not affect the detection accuracy. A low resolution dataset was used to confirm the robustness of the NPAE algorithm.

The NPAE algorithm performed better compared to traditional image data optimization methods. Image compression usually results in a loss of detection accuracy. Compared to NPAE algorithm, image compression lost 23.10% of average precision on 1920×1080 resolution, 37.94% on 1280×720, and 8.97% on 640×480. Image cropping cannot be calculated to obtain the optimal size of the cropping area, so it is difficult to accurately balance detection accuracy and detection speed.

The NPAE algorithm has four advantages: 1). It accelerates detection without loss of accuracy, 2). It can be used on a variety of computing platforms, 3). It is easily integrated with other acceleration and detection methods. 4). it does not require any training.

In future work, we will explore the use of NPAE algorithm for pedestrian image position estimation by introducing depth information. At the same time, the NPAE algorithm can also be used to accelerate pedestrian detection of multi-modal fusions.

Chapter 5

Cross-modal verification for 3D object detection

Contents

5.1 Motivation

To overcome the deficiency in the single modality of LiDAR point cloud, we propose a cross-modal verification (CMV) model for reducing 3D object detection false positives. The abundant color and texture information in image modality allow the classification of the projection region of 3D bounding box proposal in the image plane. Three 3D object detectors are adopted as backbone and eight evaluation metrics are used to fully investigate the proposed model. The experiment results show that the proposed CMV model removes more than 50% of false positives in 3D object detection proposals and improves the performance of 3D object detection.

5.2 Introduction

3D object detection is a key component of the autonomous vehicle perception system. Based on this component, the object shape and spatial position information are obtained. Furthermore, image scale variation and perspective transformation problems can be easily avoided in 3D object detection. Therefore, 3D object detection has become a key research area.

Recently, many excellent 3D detectors have been proposed. Depending on the data modality employed, these detectors can be categorized into three types. 1) LiDAR-based 3D object detection. A large number of network structures [Yan et al., 2018, Lang et al., 2019, Shi et al., 2020a] have been developed based on point feature extraction [Qi et al., 2017a] or voxel feature extraction [Zhou and Tuzel, 2018] methods. LiDAR is not affected by lighting conditions and has accurate depth information. 2) Image-based 3D object detection. There are some papers [Feng et al., 2020] that attempt to locate 3D objects on images. Depth information can be inferred from monocular, stereo or continuous frames, allowing for 3D object detection on the image. 3) Multi-modal 3D object detection. [Cui et al., 2021] shows several multi-modal fusion methods for 3D object detection. The multi-modal method extracts features from different modalities and fuses them. In this way, the robustness of the detection can be enhanced.

However, due to the difference in data modalities, the detection algorithms based on the image modality need to be modified to adapt to the point cloud modality. To overcome the shortage of feature extraction from point cloud modality, [Qi et al., 2017a] and [Zhou and Tuzel, 2018] propose the point-wise and the

voxel-wise feature extraction modules, respectively. Based on these two feature extraction modules, many LiDAR-based 3D object detectors have been proposed [Yan et al., 2018, Lang et al., 2019, Shi et al., 2020a].

For the image modality, there are two approaches to implement 3D object detection. 1) [Feng et al., 2020] attempt to locate 3D objects only using image modality. Since the depth information in the image modality is non-explicit, the detection precision of the image-based method is significantly lower than that of the LiDAR-based method. 2) The other approach is to fuse images and LiDAR. With two data modalities, it is able to detect 3D object robustly and accurately [Cui et al., 2021].

Among the mentioned three types of 3D detection, the LiDAR-based methods usually achieve the best results. This is due to the robustness of LiDAR to scale invariant and light changing. However, LiDAR point clouds are sparse and unstructured, without color or texture information of the objects. As a result, there are a large number of false positives in the LiDAR-based 3D detectors. The abundance of false positives (FPs) is a heavy burden both for autonomous driving systems and for human drivers.

Therefore, we propose a cross-model verification (CMV) model that uses image modality to filter the detection results of LiDAR-based 3D detectors. The proposed CMV model is a novel method to address result-level fusion. All 3D detection proposals are projected onto image plane. The proposed CMV model removes FPs based on image classification results. The experimental results show that the CMV model can significantly reduce the number of FPs by up to 50%.

5.3 Related Work

5.3.1 LiDAR-based 3D Object Detection

The data structure of LiDAR point cloud is irregular and orderless. A traditional CNN designed for dense image modality could not operate properly. To accommodate the data format required for CNN, some detection methods project the LiDAR point cloud into a bird's-eye view (BEV) image. In order to extract features from orderless point clouds, PointNet [Qi et al., 2017a] propose to use symmetry function. This sophisticated design has been referenced and used in many 3D object detectors. Due to the large number of LiDAR point clouds, VoxelNet [Zhou and Tuzel, 2018] proposes to use Voxel Feature Encoder (VFE) for feature extraction of the voxelized

point cloud. This kind of method usually yields better results.

5.3.2 Image-based Object Classification

Image classification is a fundamental research in computer vision where deep neural networks have achieved great success. Some milestones like the LeNet-5 [LeCun et al., 1998], AlexNet [Krizhevsky et al., 2017] and ResNet [He et al., 2016] continue to improve the deep learning-based classifier capabilities. A typical image object classifier consists of two modules: CNN-based feature extraction and class regression based on the fully connected layer.

5.3.3 Multi-modal Fusion 3D Object Detection

Multi-modal fusion is used to compensate for the instability of a single modality. LIDAR provides accurate depth information and is not affected by lighting conditions. Images can provide object color and texture detail information. The multi-modal 3D object detection methods can be categorized into early fusion, middle fusion and late fusion depending on the stage at which fusion is performed. The late fusion methods focus on fusing images with LiDAR detection results based on projection and geometric constraints.

We find that various 3D object detectors have plenty of false positives in their detection results. By adding image modality, the proposed cross-modal verification (CMV) model can easily identify and remove these false positives. The proposed CMV model is plug-and-play and is able to be integrated into various 3D object detectors to help improve detection performance.

5.4 Methods

5.4.1 Cross-Modal Verification for 3D object detection

Our proposed Cross-Modal Verification (CMV) model is developed upon the LiDAR-based 3D object detector. We attempt to design the CMV capable of adapting different detectors and reducing the presence of a large number of false positives. Therefore, CMV is a late fusion method for optimizing the detection proposals of 3D object detectors. As shown in Figure 5.1, the 3D bounding box generation module is inherited from the classic LiDAR-based 3D detector. The multi-modal fusion module

for proposal verification is built on spatial to 2D projection and image classification.

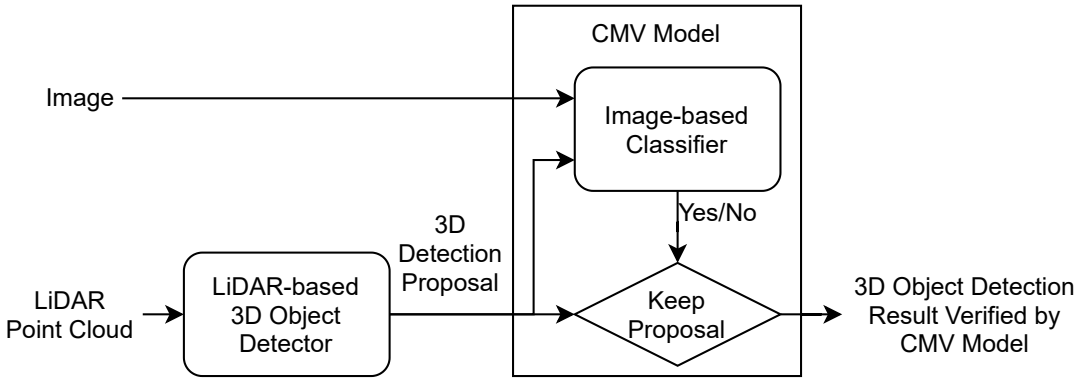


Figure 5.1: Cross-modal verification for 3D object detection.

The essential aspect of the CMV fusion model is the projection of the LiDAR point cloud onto the image plane. For a spatial point $\mathbf{pt}_{pc} = (x, y, z)$, there is the projected point $\mathbf{pt}_{img} = (u, v)$ in the image plane. The projection point calculation process includes affine transformation and coordinate system transformation. Here we introduce the projection method using the KITTI 3D object detection dataset [Geiger et al., 2013] as an example. The conversion can be obtained from Equation 5.1.

$$\mathbf{pt}_{img} = \mathbf{P}^{(i)}_{rect}\mathbf{R}^{(0)}_{rect}\mathbf{pt}_{pc} \tag{5.1}$$

Where $\mathbf{P}^{(i)}_{rect} \in \mathbb{R}^{3\times4}$ is the projection matrix after rectification. $\mathbf{R}^{(0)}_{rect} \in \mathbb{R}^{4\times4}$ is the expanded rectifying rotation matrix. i is the camera index.

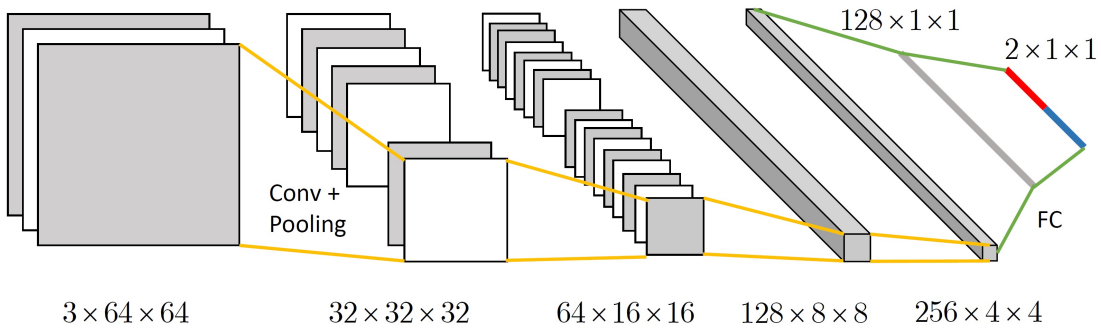


Figure 5.2: Image-based classifier for verification in CMV model.

In this way, we obtain the 3D bounding box and its projection on the image plane. All that remains is to prepare an image-based classifier for verification. As

shown in Figure 5.2 we design a six layer convolutional neural network with reference to LeNet. The classifier of CMV model is begin with four convolution layers, followed by two fully connected layers. The convolution kernel size is 3×3 with a 2×2 maxpooling. The input image is resized to 64×64 pixels. After four convolution layers, the output feature map size is 4×4 pixels with the batch size of 256. The first fully connected layer reduces the number of features from $4 \times 4 \times 256$ to 128. Then the last fully connected layer output the scores of two classes. Finally, the classifier outputs confidence score of classes and verifies them with 3D bounding box proposals.

The output of the class regression for the cascaded two fully connected layers are '**Foreground**' and '**Background**'. The class '**Foreground**' includes *Cars*, *Pedestrians* and *Cyclists*. The class '**Background**' consists of streets, skies, trees and road signs. In order to train an effective classifier, we collect an Autonomous Vehicle Object Recognition (AVOR) dataset.

5.4.2 Autonomous Vehicle Object Recognition Dataset

The AVOR dataset is built based on the KITTI 3D Object Detection dataset. According to the 3D bounding box from the ground truth, we crop and keep the object areas in the images as the **Foreground** class elements. To augment the data, the image area is acquired four times after random offset according to ground truth. We control the random range to the overlap area between the generated image area and the ground truth bounding box is no less than 0.7. Some special objects are filtered out according to one criterion : $max(h,w) > 15$ pixels while $min(h,w) > 10$ pixels where h and w are the height and width of the object images.

There are six classes in KITTI 3D object detection dataset. The most mentioned classes are *Car*, *Pedestrian* and *Cyclist*. These three classes are considered as the **Foreground** class in our AVOR dataset. For the **Background** class, We avoid object areas to obtain background images from KITTI dataset by random cropping. There are 114342 **Foreground** class images and 94052 **Background** class images for training. As for testing, there are 122198 **Foreground** class images and 99676 **Background** class images. In particular, we add *roadblocks* to the **Background** category. The Figure 5.3 gives an example of AVOR dataset.

Figure 5.3: Samples of AVOR dataset.

5.5 Experiments

5.5.1 Methodology

LiDAR modality based 3D object detection

Three representative detectors are selected as the baseline models for evaluation, which are SECOND [Yan et al., 2018], PointPillars [Lang et al., 2019], and PartA2 [Shi et al., 2020a]. SECOND network is a voxel-based 3D object detector that improves the sparse 3D convolution operation. PointPillars network is a pillar-based 3D object detector that introduces pillars to represent point cloud features. PartA2 network is an advanced voxel-based 3D detector which is adopted to examine whether a large number of false positives are also present.

There are 7481 training frames and 7518 test frames with both modalities in KITTI 3D object detection dataset. The ground truth of training data are available for public access. Therefore, we only used the training data and split it into two dataset for training and testing according to [Yan et al., 2018]. Since the proposed CMV model is plug-and-play, these detectors can be added directly to obtain performance improvements with minimal modifications. PointPillars network and PartA2 network are trained for 160 epochs, and SECOND network is trained for 80 epochs.

Image modality based Classifier

In training the classifier, we randomly initialized the convolution kernel parameters. Since the fully connected layer requires a fixed-length input tensor, we investigate various sizes of input images. The best performance is achieved when the input object proposal image is resized to 64×64 pixels. The activation function is replaced from sigmoid to relu. The classifier is trained for a total of 40 epochs. After

training, the classifier is integrated into the 3D object detector to compose the CMV model.

Evaluation Metrics

We present the change in the number of false positives (FP) present in the 3D object detection results after CMV model optimization. To be classified as a TP, the detection proposal bounding box should have at least an intersection over union (IoU) with the ground truth bounding box more than 0.7.

In addition, changes in true positives (TP) and false negatives (FN) are also used as performance metrics. On this base, we calculate the precision P and recall R by using Equation 5.2.

$$\begin{aligned} P &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \\ R &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \end{aligned} \tag{5.2}$$

We also present the F1 score and F2 score for evaluating our CMV model. The F1 score is a balance of precision and recall, which is better than relying on only a single metric. The F2 score puts more emphasis on recall than precision to demonstrate the changes brought about by our model. The formula for the F1 score and F2 score are given in Equation 5.3.

$$\begin{aligned} F_1 &= \frac{2PR}{P+R} \\ F_2 &= \frac{5PR}{4P+R} \end{aligned} \tag{5.3}$$

In addition to focusing on FP changes, we also calculate the average precision (AP) score. The official AP calculation method of KITTI dataset is the 40-point interpolation. Furthermore, we also applied the enhanced 40-point interpolation method introduced in Chapter 3 to calculate AP.

5.5.2 Experiment Results

Table 5.1, 5.2 and 5.3 show the scores of various metrics for 3D object detection of the CMV model with SECOND, PointPillars and PartA2 backbone, respectively. In all three backbone networks, a large number of false positives exists. After the processing of the CMV model, there is a significant reduction in the number of FPs. For example, the CMV model reduces FP by 45.87%, 42.40% and 23.99% in the class *car*, respectively.

Table 5.1: Results of CMV model with SECOND backbone.

Class	Model	TP	FP	FN	PRE	REC	F1	F2	AP	eAP
CAR	SECOND	6611	6999	1175	48.57	84.90	61.79	73.85	78.21	77.63
	SECOND-CMV	6546	**3788**	1241	**63.34**	84.06	**72.24**	**78.90**	76.91	77.04
PED	SECOND	1220	3363	519	26.62	50.93	34.96	43.06	57.33	57.93
	SECOND-CMV	1146	**1549**	593	**42.52**	48.01	**45.10**	**46.80**	55.24	55.50
CYC	SECOND	456	2158	88	17.44	27.95	21.48	24.95	64.86	66.04
	SECOND-CMV	422	**795**	122	**34.67**	25.37	**29.30**	**26.81**	61.53	62.51

Table 5.2: Results of CMV model with PointPillars backbone.

Class	Model	TP	FP	FN	PRE	REC	F1	F2	AP	eAP
CAR	PointPillars	6423	7367	1366	46.57	82.46	59.53	71.45	74.61	74.23
	PointPillars-CMV	6348	**4243**	1441	**59.93**	81.49	**69.07**	**76.02**	72.85	72.36
PED	PointPillars	1126	6555	613	14.65	45.18	22.13	31.89	45.06	45.89
	PointPillars-CMV	1071	**2252**	668	**32.22**	42.63	**36.70**	**40.04**	46.39	46.70
CYC	PointPillars	405	2144	137	15.88	22.86	18.75	21.02	60.29	61.19
	PointPillars-CMV	383	**721**	160	**34.69**	20.99	**26.16**	**22.79**	58.29	58.81

Table 5.3: Results of CMV model with PartA2 backbone.

Class	Model	TP	FP	FN	PRE	REC	F1	F2	AP	eAP
CAR	PartA2	6770	5142	1027	56.83	86.82	68.69	78.53	82.65	81.91
	PartA2-CMV	6749	**3908**	1048	**63.32**	86.55	**73.14**	**80.64**	82.74	81.75
PED	PartA2	1256	4109	483	23.41	55.01	32.84	43.31	53.46	54.11
	PartA2-CMV	1216	**2146**	523	**36.16**	53.71	**43.22**	**48.96**	53.50	53.72
CYC	PartA2	468	1060	75	30.62	31.30	30.96	31.16	71.27	71.44
	PartA2-CMV	454	**587**	89	**43.61**	30.22	**35.70**	**32.20**	70.22	70.24

In principle, the CMV model is designed to remove FPs from detection proposals, which means that the number of TPs cannot be increased. Thus a general conclusion can be drawn by examining the precision scores and recall scores. Benefits from the significant reduction of FP, the detection accuracy has been greatly improved. Yet, the recall scores without FP in the calculations are slightly reduced.

For a more comprehensive evaluation of the CMV model, we need to refer to the F1 score metric. The three baseline networks achieve significant improvements in F1 score in all classes of object. It implies that the reduction of FPs brings a change in precision and recall that is beneficial to detectors. Even for the F2-score, where recall is more emphasized, all results indicate an improvement of detection.

We notice that even false positives have been removed more than half, the aver-

age precision do not improve significantly. The reason for this evaluation result is that the average precision does not consider FP to be as important as TP. In the process of calculating the average precision, all detection results are sorted by the given class confidence scores. Then N interpolation intervals are applied to integrate the area under curve (AUC) of the precision-recall (PR) curve. Due to the low confidence scores of most FPs, they only have an impact in the last interpolation interval, which accounts for just 1/N of the overall average precision. At the same time, the reduction of TP may lead to a decrease in the number of interpolation intervals. This eventually leads to a significant degradation of average precision. Therefore, since the average precision calculation takes TP more importantly, it leads to the results in tables.

5.6 Conclusion

In this chapter, a cross-modal verification (CMV) model was proposed for reducing 3D object detection false positives. The extensive color and texture information in the image modal was used to complement the deficiencies of the LiDAR point cloud modal. The proposals obtained from the 3D object detector were verified by the CMV model and the false positives were discarded. To train the CMV model, an Autonomous Vehicle Object Recognition dataset was built. Three 3D object detectors and nine metrics were adopted to fully investigate the proposed model. All experimental results showed the enhancement of the proposed CMV model for 3D object detection.

Our future work will focus on embedding CMV model into 3D object detectors to obtain an end-to-end architecture. In this way, the embedded model can verify directly at the 3D region proposal module.

Chapter 6

AF3D: Asynchronous Multi-Modal Fusion for 3D Object Detection

Contents

6.1 Motivation

Increasing the number of fusion modalities is a common procedure for obtaining robust 3D object detection. The synchronization of all modalities becomes a fundamental assumption, but its limitations are neglected. It has been observed from many multi-modal datasets that the data frame rate decreases significantly due to synchronization. To address this problem, we propose an Asynchronous multi-modal Fusion 3D object detection (AF3D) model. The proposed AF3D is able to perform object detection for both synchronous and asynchronous frames. Since the multi-modal data in asynchronous frames are incomplete, they are transformed into point cloud modality for uniform detection. In particular, AF3D performs asynchronous multi-modal fusion on two consecutive frames, which could counteract the degradation of detection precision due to modal incompleteness. Experimental results show that AF3D can detect 3D objects using a single detector for both synchronous and asynchronous frames. Notably, the average precision score for monocular image modality is improved by 9.4% with asynchronous multi-modal fusion method.

6.2 Introduction

3D object detection is a fundamental task for autonomous vehicles to accurately perceive the environment. Recently, LiDAR-based 3D object detection have achieved remarkable progress. How to make the detector more robust becomes an interesting topic. Intuitively, increasing the data modalities for multi-modal fusion is a promising approach. LiDAR modality can accurately describe spatial depth. Image modality is rich in semantic information. The two modalities are complementary and can be used jointly to improve the detection performance. Many existing multi-modal fusion methods are attempting to develop an effective fusion paradigm [Feng et al., 2020].

An underlying assumption of most fusion architectures [Sindagi et al., 2019, Pang et al., 2020, Zhang et al., 2021c] is that all modalities have been well synchronized. Since data in asynchronous frames are discarded, the frequency of synchronized data is usually lower than the sampling frequency of sensors. This is a factor which limits the detection efficiency. It cannot be ignored in addition to the detector design itself. This degradation of data rate due to synchronization is common in

most multi-modal datasets as shown in Table 6.1.

Table 6.1: The synchronous sampling frequency of multiple modal sensors reduces the flow rate of data. **Syn** indicates the synchronous frame rate.

	Camera	LiDAR	Radar	**Syn. Frame**
KITTI	15 fps	10 fps	-	**10 fps**
A2D2	30 fps	20 fps	-	**10 fps**
nuScenes	12 fps	20 fps	13 fps	**2 fps**

In KITTI [Geiger et al., 2013], A2D2 [Geyer et al., 2020] and Waymo [Sun et al., 2020], two modalities are provided. Even though some sensors could sample at 30 Hz, the synchronized frame rate drops to 10 Hz. If there are more modalities to enroll, the situation becomes even more critical. As reported in nuScenes [Caesar et al., 2020], radar modality is introduced in addition to image and LiDAR. Along with this, the synchronization frame rate is reduced to 2 Hz. Under such circumstances, it is necessary to restrict the speed of AVs to maintain safety.

One possible solution to address the low synchronization frequency is to upgrade all sensors to the same sampling frequency. Apparently, it will take a lot of effort and a long time to make progress. Therefore, we intend to improve the fusion algorithm based on the existing sensors to get rid of the limitation of low sampling frequency. According to Table 6.1, the sampling frequency of the LiDAR is normally slower than that of the camera. As a result, there are multiple asynchronous image frames captured between the two synchronous frames. Exploiting these asynchronous image frames is a possible solution to the problem.

An intuitive approach is to perform 3D object detection on both synchronous and asynchronous frames directly. By doing that, it brings several consequences. Existing fusion models are unable to work properly in a single modality situation. Different modalities demand unique feature extractors, which means that multiple networks are required to cope with the transitions between synchronous and asynchronous frames.The complex architecture put a heavy burden on the AVs. Furthermore, there is a huge gap between image-based and LiDAR-based detectors. It needs efforts to balance the difference between these two modalities. Otherwise, periodic fluctuations in detection accuracy could undermine the safety of AVs.

A novel architecture, Asynchronous multi-modal Fusion 3D object detector (AF3D), is proposed to unify the detection of synchronous and asynchronous frames. By introducing the 3D motion, the asynchronous frames could be fused

with the synchronous frames. This asynchronous fusion allows embedding information from synchronous frames into asynchronous fusion. It improves the 3D object detection accuracy in asynchronous image modality frames. In addition, the image modality could be transformed into LiDAR modality by asynchronous fusion. In this case, only one unified LiDAR-based 3D object detector is required for both synchronous and asynchronous frames. To the best of our knowledge, this is the first research on the fusion of synchronous and asynchronous frames for 3D object detection. The experiment results show that our AF3D outperforms the current state-of-the-art monocular image 3D object detector. At the same time, the high detection accuracy of the LiDAR modality is preserved.

6.3 Related Work

3D object detection based on two modalities can be summarized into three categories. 1) Image-based 3D detection. 2) LiDAR-based 3D detection. 3) Fusion-based 3D detection. Each method has its capabilities. However, LiDAR-based 3D detection has been leading in benchmarks [Geiger et al., 2013, Sun et al., 2020]. A brief introduction of these methods is presented below.

1) Image-based 3D detection. Some image-based methods [Li et al., 2019b, Li et al., 2020] build the 3D object detection based on 2D features. By modeling perspective constraints, these methods estimate the 3D spatial position of an object from its 2D position in the image. Other methods [Wang et al., 2019, Peng et al., 2021] extract 3D features directly from the image and predict the spatial location of the object. None of these methods could avoid errors since vital depth information is lost in the image.

2) LiDAR-based 3D object detection

The point cloud sampled by LiDAR is sparse and disordered. It is a completely different data structure compared to images. The grid-based methods [Yan et al., 2018, Lang et al., 2019] first slice the 3D space into dense cubes. The CNN network is then adopted to extract features and regress the object locations. The point-based methods [Qi et al., 2017a, Qi et al., 2018] first output feature maps using a rotation-insensitive feature extractor, followed by an object location regression. Since LiDAR has accurate depth information and insensitive to light, the LiDAR-based 3D object detectors achieve state-of-the-art performance.

3) Fusion-based 3D detection

Despite the great performance achieved by LiDAR-based detectors, it still lacks semantic information from image modality. As a result, certain fusion-based detectors [Sindagi et al., 2019, Pang et al., 2020] are proposed to improve the robustness. The feature-level fusion approach combines the feature maps of both modalities. The result-level fusion approach cross-validates the 3D detection results between the two modalities. As we can note that all fusion-based 3D detectors require the synchronized data frame. Moreover, a complex architecture including different feature extraction and feature fusion is always necessary.

There is a lack of experience in integrating asynchronous frames into a synchronous frame-based detection model. We would like to contribute a unified detection model AF3D for both asynchronous and synchronous frames with multiple modalities. In our model, synchronous frames should be used to improve the detection of asynchronous image modality frames.

Pseudo-LiDAR [Wang et al., 2019] provides a possible path to build a unified detection model. The image-based depth map is converted to the point cloud named pseudo-LiDAR. In this case, all existing LiDAR-based detectors can be applied for 3D object detection, regardless of the modality. We design an asynchronous multi-modal fusion method based on 3D motion estimation with reference to pseudo-LiDAR. The image modality is transformed into LiDAR modality for unified 3D object detection. By estimating the 3D motion, the connection between two consecutive frames is established. The accurate depth information of LiDAR modality in synchronous frames contributes to the 3D detection of image modality in asynchronous frames. The design details of the AF3D model are presented in the next chapter.

6.4 Asynchronous Multi-Modal Fusion based Modal Transformation

The data in the 3D object detection discussed in this thesis alternates between synchronous and asynchronous. There are two modalities in the synchronous frame, which are image and LiDAR point cloud (referred by LiDAR). In the asynchronous frame, there is only the image modality. This is a common observation, as the camera sampling frequency is usually higher than that of LiDAR.

In our AF3D model, LiDAR is set as the default modality for performing the 3D

object detection. By using asynchronous multi-modal fusion, the LiDAR modality of asynchronous frame can be created jointly with synchronous frame. In this way, the image modality of the asynchronous frame also has a corresponding LiDAR modality. Any existing 3D object detector backbone can be applied. There are three modules in the asynchronous multi-modal fusion architecture as shown in Figure 6.1.

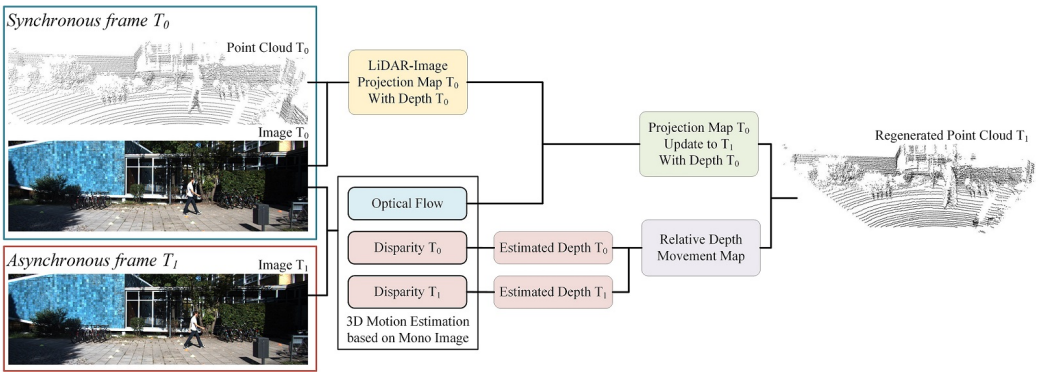


Figure 6.1: AF3D asynchronous fusion framework. The LiDAR modality of point cloud for the asynchronous frame is generated by fusing the image and LiDAR of the synchronous frame with the image of the asynchronous frame.

Module I: 3D points projection to image plane

In this module, the 3D point $\mathrm{P}_w^0 = (\mathrm{X}_w^0, \mathrm{Y}_w^0, \mathrm{Z}_w^0)$ in the world coordinate system acquired by LiDAR is projected onto the image plane with the depth attached to get the point $p_d^0 = (u^0, v^0, d^0)$.

First, the world coordinate system of the 3D point P_w needs to be converted to the point $\mathrm{P}_c^0 = (\mathrm{X}_c^0, \mathrm{Y}_c^0, \mathrm{Z}_c^0)$ with the camera coordinate system. These two coordinate frames are related via a rotation and a translation. Equation 6.1 shows the transformation. R is a rotation matrix of size 3×3. t is a translation 3-vector which move the origin coordinate to the center of image coordinate.

$$\begin{pmatrix} \mathrm{X}_c^0 \\ \mathrm{Y}_c^0 \\ \mathrm{Z}_c^0 \\ 1 \end{pmatrix} = \begin{bmatrix} \mathrm{R} & -\mathrm{R}t \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix} \begin{pmatrix} \mathrm{X}_w^0 \\ \mathrm{Y}_w^0 \\ \mathrm{Z}_w^0 \\ 1 \end{pmatrix} \tag{6.1}$$

Then, the 3D point P_c^0 is projected onto the image plane to obtain the point $p_c^0 = (x_c^0, y_c^0)$. The center of the image plane is the origin coordinate of the camera coordinate system. The projection could be derived from Equation 6.2. f is the

focal length of the camera.

$$\begin{aligned} x_c^0 &= f\frac{\mathrm{X}_c^0}{\mathrm{Z}_c^0} \\ y_c^0 &= f\frac{\mathrm{Y}_c^0}{\mathrm{Z}_c^0} \end{aligned} \tag{6.2}$$

Equation 6.2 could be rewritten with homogeneous coordinate as shown in Equation 6.3.

$$\begin{pmatrix} x_c^0 \\ y_c^0 \\ 1 \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} \mathrm{X}_c^0 \\ \mathrm{Y}_c^0 \\ \mathrm{Z}_c^0 \\ 1 \end{pmatrix} \tag{6.3}$$

Then, the point $p_c^0 = (x_c^0, y_c^0)$ in camera coordinate is converted to the pixel point $p_i^0 = (u_i^0, v_i^0)$ in image coordinate. This process involves converting distances to pixels and moving the origin coordinate to a corner of the image. The transformation from the 3D point P_c^0 to the image coordinate point p_i^0 is shown in Equation 6.4. k_x and k_y are the number of pixels per unit distance in the x and y directions in image coordinate. C = (c_x, c_y) is the center point in the image coordinate.

$$\begin{pmatrix} u_i^0 \\ v_i^0 \\ 1 \end{pmatrix} = \begin{bmatrix} k_x f & 0 & c_x & 0 \\ 0 & k_y f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} \mathrm{X}_c^0 \\ \mathrm{Y}_c^0 \\ \mathrm{Z}_c^0 \\ 1 \end{pmatrix} \tag{6.4}$$

Finally, the depth d_c^0 of the 3D point P_c^0 in camera coordinate is attached to p_i to obtain $p_{id}^0 = (u_i^0, v_i^0, d_c^0)$.

Module II: Build pixel-level 3D motion field

The 3D motion field represents the movement of points in two consecutive frames. Jointly the Joining the image modalities of synchronous and asynchronous frames, the 3D motion field can be inferred. The estimation of 3D motion field includes optical flow estimation and depth estimation.

Optical flow estimation is a method that detects the intensity of image pixels over time to infer the speed and direction of object movement. With the optical flow, the displacement $m = (\xi_u, \xi_v)$ at each pixel of the previous frame is obtained.

Depth estimation computes the depth information from the disparity map. A

disparity map calculation algorithm takes a pair of left-right images I_l and I_r as input. These images are captured from a pair of cameras with a horizontal offset b. The output disparity map $\Delta(u, v)$ of each pixel locations is the same size as the input image. Equation 6.5 is the formula that converts the disparity map $\Delta(u, v)$ to the depth map D.

$$D = \frac{f \times b}{\Delta(u, v)} \tag{6.5}$$

Usually stereo images are required for scene flow to calculate disparity map. While, it is possible for deep neural network infers disparity map directly from monocular images. The [Hur and Roth, 2020] is a representing network to build 3D motion field from monocular consecutive images. They design a CNN network to estimate depth and 3D motion simultaneously from a optical flow cost volume. Self-supervised learning is used to leverage unlabeled data, which includes a 3D loss functions and occlusion reasoning. We integrate this well-trained network directly into our AF3D model for 3D motion field estimation.

Module III: Update point cloud with relative movement

First, together with the point p_i^0 of the previous frame and the optical flow m, the point of the current frame $p_i^1 = (u_i^0 + \xi_u, v_i^0 + \xi_v) = (u_i^1, v_i^1)$.

Next, the depth information needs to be recovered. In pseudo-LiDAR [Wang et al., 2019], the depth map is used directly to generate point cloud. Yet, the depth error is discontinuous and thus results in a very significant dispersion of pseudo-LiDAR. For instance, for a 5-meter-away object, a unit disparity error(in pixels) implies a depth error of 10 cm. However, for an object 50 meters away, the same disparity error brings a depth error of 5.8 m [Ma et al., 2020].

For the above reason, we use the relative depth to build depth map instead of using estimation depth directly, as shown in Equation 6.6.

$$d_c^1 = d_c^0 + (D^1 - D^0) \tag{6.6}$$

Where D^1 is the depth estimated from asynchronous frame image. D^0 is the depth estimated from synchronous frame image. d_c^0 is the accurate depth information from synchronous frame LiDAR. d_c^1 is the restored point cloud of asynchronous frame.

By using Equation 6.7 with camera intrinsic, the point $P_c^1 = (X_c^1, Y_c^1, Z_c^1)$ in image coordinate for asynchronous frame is calculated.

$$
\begin{aligned}
\mathrm{Z}_c^1 &= d_c^1 \\
\mathrm{X}_c^1 &= \frac{(u_i^1 - c_u) \times \mathrm{Z}_c^1}{f_u} \\
\mathrm{Y}_c^1 &= \frac{(v_i^1 - c_v) \times \mathrm{Z}_c^1}{f_v}
\end{aligned}
\tag{6.7}
$$

Where$(c_u, c_v)$is the image center point. f_u and f_v are the horizontal and vertical focal lengths. It should be reversed from image coordinate to world coordinate for LiDAR-based 3D object detector.

6.5 Experiments and Results

6.5.1 Dataset and backbone network

KITTI benchmark provides variety annotations for different tasks, such as disparity, optical flow, scene flow and 3D object detection. Many advanced algorithms are submitted for comparison and can be integrated into our AF3D model. We choose to validate the effectiveness of our algorithm on the KITTI dataset.

Since asynchronous multi-modal fusion is a novel fusion paradigm, it is necessary to build a suitable dataset for training and testing. Three sub-datasets are involved in the construction of the Asynchronous KITTI dataset. We retrieve the previous frame of each KITTI 3D object detection dataset frame from the KITTI raw dataset. As a result, each frame of Asynchronous KITTI dataset contains two images and two LiDAR point clouds. This allows us to directly use the labels and calibrations of the KITTI 3D object detection dataset. In addition to frame number 5524, there are 7480 frames in the Asynchronous KITTI dataset. KITTI scene flow dataset is also applied to train scene flow estimator for deriving 3D motion. The splitting approach for the training and testing datasets is referred to [Yan et al., 2018].

1) Asynchronous multi-modal fusion model. In this model, we equip the scene flow network [Hur and Roth, 2020] to estimate 3D motion. It is a monocular scene flow network based on self-supervised learning. It can be deployed on the KITTI dataset and provides satisfactory performance. The well-trained weight is directly integrated into our AF3D. The scene flow network can be replaced arbitrarily according to the requirements.

2) Unified LiDAR-based 3D object detector. The PointPillars [Lang et al., 2019]

network is employed for detection. It is an improved voxel-based network which achieves convincing accuracy and efficiency. Since we transform the image modality to LiDAR modality, the 3D object detector can also be selected arbitrarily. We train the PointPillars with our asynchronous multi-modal dataset. First, the scene flow network is used to generate LiDAR point cloud for asynchronous frames. Second, the 3D object detector is trained on the generated LiDAR for 160 epochs. This weight is used for asynchronous frame 3D object detection. We also use the original LiDAR point cloud to train a weight. It is used for 3D object detection in synchronous frames.

6.5.2 Metric

The metric used to evaluate the accuracy of 3D object detection is the average precision. For the KITTI 3D object detection dataset, the average precision is calculated by the 40-point interpolation method [Simonelli et al., 2019]. The evaluation procedure is shown in Equation 6.8.

$$\begin{aligned} \mathrm{AP_N} &= \frac{1}{\mathrm{N}} \sum_{\mathrm{R} \in \mathbb{R}_\mathrm{N}} \mathrm{P}_{max}(\mathrm{R}) \\ \mathrm{P}_{max}(\mathrm{R}) &= \max_{\mathrm{R}':\mathrm{R}' \geq \mathrm{R}} \mathrm{P}(\mathrm{R}') \end{aligned} \tag{6.8}$$

Where $\mathbb{R}_\mathrm{N}$ is the set of N interpolation points. Specifically, $\mathbb{R}_{40} = \{1/40, 2/40, ..., 40/40\}$ for 40-point interpolation method.

In our previous study [Zhang et al., 2022], 40-point interpolation method may lead to confusion due to average precision distortion. Therefore, we also use the enhanced 40-point interpolation method to evaluate the average precision AP_h of the detector. The evaluation procedure of $\mathrm{AP}_{e\mathrm{N}}$ is shown in Equation 6.9.

$$\begin{aligned} \mathrm{AP}_{e\mathrm{N}} &= \frac{1}{\mathrm{N}} \sum_{i=1}^{\mathrm{M}-1} \mathrm{P}_{max}(\mathrm{R}^i_{ip}) + \mathrm{T} \times \mathrm{P}(\mathrm{R}^d_{ip}) \\ \mathrm{P}_{max}(\mathrm{R}^i_{ip}) &= \max_{\mathrm{R}':\mathrm{R}' \geq \mathrm{R}^i_{ip}} \mathrm{P}(\mathrm{R}') \end{aligned} \tag{6.9}$$

The number of valid interpolation intervals M and the middle interpolation points R^i_{ip} are calculated as follows. $\lceil * \rceil$ means rounding up.

$$\begin{aligned} \mathrm{M} &= \lceil \frac{\mathrm{N}_{tp} \times \mathrm{N}}{\mathrm{N}_{gt}} \rceil \\ \mathrm{R}_{ip}^{i} &= \frac{2i+1}{2\mathrm{N}}, i \in [0, 1, \ldots, \mathrm{N}-1] \end{aligned} \tag{6.10}$$

The valid interval width $\mathrm{T} = \mathrm{N}_{ltp}/\mathrm{N}_{gt}$. The number of true positives for the last valid interpolation interval N_{ltp} and the dynamic interpolation point for the last interpolation interval R_{ip}^{d} can be acquired as follows. $\lfloor * \rfloor$ means rounding down.

$$\begin{aligned} \mathrm{N}_{ltp} &= \mathrm{N}_{tp} - \lfloor \frac{\mathrm{N}_{gt}}{\mathrm{N}} \times (\mathrm{M}-1) \rfloor \\ \mathrm{R}_{ip}^{d} &= \frac{\lceil \frac{\mathrm{N}_{ltp}}{2} \rceil + \lfloor \frac{\mathrm{N}_{gt}}{\mathrm{N}} \times (\mathrm{M}-1) \rfloor}{\mathrm{N}_{gt}} \end{aligned} \tag{6.11}$$

6.5.3 Experimental result

Table 6.2 shows the 3D object detection results for monocular images. The pseudo-LiDAR [Wang et al., 2019] (PL-AVOD, PL-FPN) represents the modality transform without asynchronous multi-modal fusion. The LPCG [Peng et al., 2021] (LPCG-RTM3D, LPCG-MonoFlex) represents the existing state-of-the-art in monocular image-based 3D detector. Our AF3D with the PointPillars backbone achieves the best detection accuracy.

Table 6.2: Comparison of AF3D with other state-of-the-art detectors for monocular 3D object detection results. E: Easy, M: Moderate, H: Hard. 3D@0.5: 3D object detection with IoU threshold 0.5. 3D@0.7: 3D object detection with IoU threshold 0.7.

Model	3D@0.5			3D@0.7		
	E	M	H	E	M	H
PL-AVOD	57.0%	42.8%	36.3%	19.5%	17.2%	16.2%
PL-FPN	66.3%	42.3%	38.5%	28.2%	18.5%	16.4%
LPCG+RTM3D	65.44%	49.40%	43.55%	25.23%	19.43%	16.77%
LPCG-MonoFlex	69.16%	54.27%	48.37%	31.15%	23.42%	20.60%
AF3D	**76.55**%	**57.19**%	**52.99**%	**35.36**%	**25.63**%	**22.61**%

This result indicates that asynchronous multi-modal fusion is an effective method for image modality transformation. It helps our AF3D network to improve the average precision of 3D object detection for monocular image by 9.4% (2.21% in AP score) compared to the state-of-the-art method (Moderate, IoU threshold is 0.7).

Table 6.3 illustrates the full capabilities of AF3D for 3D object detection on both synchronous and asynchronous frames. For the class '*car*' with an IoU threshold of

Table 6.3: 3D object detection results of AF3D in the image modality and the LiDAR modality.

Mono Image Modality (AP/AP$_{eN}$)			
	Easy	Moderate	Hard
Car@0.5	78.85%/77.98%	57.19%/57.11%	52.98%/50.99%
Car@0.7	35.58%/32.50%	25.63%/21.38%	22.61%/18.36%
Pedestrian@0.25	21.47%/16.91%	19.14%/15.24%	17.42%/12.95%
Pedestrian@0.5	6.52%/4.45%	5.95%/3.40%	5.71%/2.77%
Cyclist@0.25	4.20%/2.42%	3.81%/1.84%	3.75%/1.67%
Cyclist@0.5	3.03%/0.39%	3.03%/0.24%	3.03%/0.26%
LiDAR Modality			
	Easy	Moderate	Hard
Car@0.5	96.62%/97.13%	94.77%/94.01%	92.29%/91.34%
Car@0.7	86.61%/86.06%	74.61%/74.23%	70.23%/69.89%
Pedestrian@0.25	70.01%/71.06%	66.40%/67.29%	62.21%/63.37%
Pedestrian@0.5	50.54%/51.31%	45.06%/45.89%	40.01%/41.17%
Cyclist@0.25	85.81%/86.98%	71.19%/72.02%	67.26%/68.21%
Cyclist@0.5	80.80%/81.35%	60.29%/61.19%	56.27%/57.45%

0.5, the detection accuracy gap between the two modalities has been significantly narrowed. However, for a higher IoU threshold as 0.7, it still requires a further improvement.

Detection results for classes '*pedestrian*' and '*cyclist*' are also provided. The difference in average precision scores between the two modalities is enormous. These objects have a smaller volume and contour compared to the '*car*'. When generating depth map from the image of asynchronous frame, estimation errors could easily corrupt the integrity of the object geometry.

Figure 6.2 shows two samples of point cloud generated by AF3D based on monocular images. Compared to ground truth, the basic contour of the object remains after moving the synchronous frame point cloud according to the 3D motion. It contributes to improve the detection accuracy.

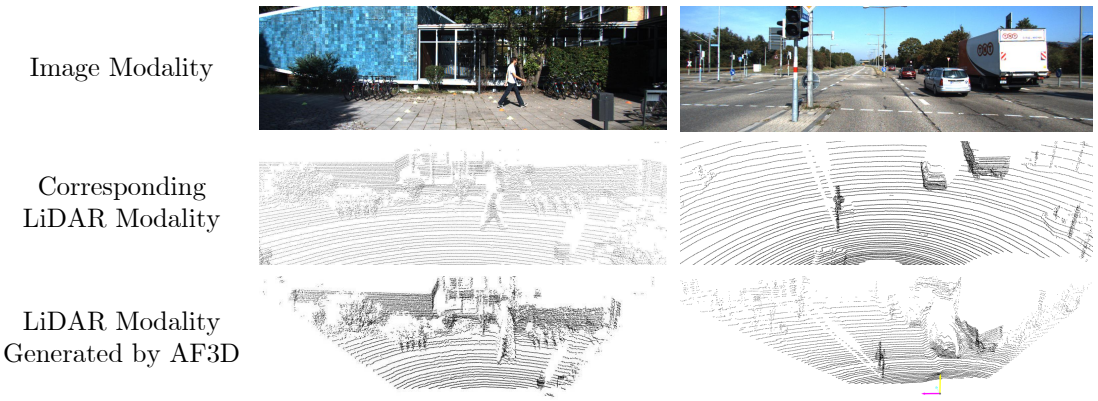


Figure 6.2: Point clouds generated by the asynchronous fusion of AF3D compared with ground truth. The first line is the image of the asynchronous frame. The second line is the point cloud ground truth. The third line is the generated point cloud.

6.6 Conclusion

In this chapter, an asynchronous multi-modal fusion 3D object detector (AF3D) was proposed. AF3D adopted scene flow network to estimate 3D motion and built the point cloud by fusing synchronous frame for the asynchronous frame. As a result, the image modality was transformed to the LiDAR modality, enabling a unified 3D object detection. Our AF3D achieved a state-of-the-art average precision score for monocular image 3D object detection. At the same time, it remained capable of detecting objects in LiDAR modality with high accuracy.

AF3D network still has some defects. In our future work, we will try to incorporate constraints between spatial points to reduce the errors of asynchronous multi-modal fusion. We also plan to merge the weights of asynchronous and synchronous detectors to achieve efficient detection.

Chapter 7

Conclusions and future work

In this thesis, we explored the 3D object detection from four aspects. The research covers evaluation metric, dataset optimization, synchronous multi-modal fusion and asynchronous multi-modal fusion for 3D object detection.

In Chapter 3, we disclosed the average precision distortion problem during the evaluation of 3D object detection performance. This distortion problem may incorrectly evaluate the performance of the detector. The enhanced N-point interpolation method was proposed in order to address the distortion problem.

In Chapter 4, we attempt to build a visual model to optimize the image modality for the dataset. Unlike frame-by-frame image optimization, the proposed NPAE algorithm establishes uniform optimization parameters for image modality based on data acquisition parameters. And it achieves acceleration on both CPU and GPU platforms.

In Chapter 5, we investigated the multi-modal fusion 3D object detection in the presence of synchronized data. In examining the advanced LiDAR-based 3D object detectors, a significant quantity of false positives were commonly presented. The cross-modal verification (CMV) model was proposed to remove the FPs by using image-modality classifier for 3D detection. The experimental results showed that more than 50% of the false positives were eliminated and the 3D detection results were improved.

In Chapter 6, an asynchronous multi-modal fusion for 3D object detection (AF3D) was exposed for the first time. We transformed the image modality into a LiDAR modality in order to use the LiDAR-based 3D object detector for both asynchronous and synchronous frames. To address the huge gap in the 3D detection performance of the two modalities, an asynchronous fusion module was designed. The spatial motion information obtained from the two image frames was used to generate asynchronous frame point clouds by shifting the synchronous frame point clouds. After asynchronous fusion, the 3D detection accuracy of image modalities was improved by 9.43%. It outperformed the state-of-the-art monocular image modal 3D detectors.

In this paper, the multi-modal fusion 3D object detection algorithms based on deep learning is investigated. Despite the results of the study, there is still a lot of room for improvement. In that case, my future research will focus on three areas: 1) The Advanced fusion architecture. Recently it has been noticed that new fusion algorithms are being proposed, such as BEVFusion [Liu et al., 2022] and Transformer fusion [Yan et al., 2023]. The accuracy of 3D object detection based on the novel fu-

sion paradigm outperforms that of the single modality. The effectiveness of fusion is demonstrated. This proves that it is essential to continue exploring a more effective fusion architecture. 2) Fusion with radar modality. Radar modality is rarely fused with other modalities for autonomous vehicles. Due to its modal characteristics, only the information of the moving target is preserved. This requires some changes in the neural network that processes the radar signal. A type of neural network called Spiking Neural Network (SNN) [Tavanaei et al., 2019] could be used to process radar modality [Viale et al., 2021]. How to effectively fuse the radar modality and how to fuse it with novel neural network structures is an interesting challenge. 3) Lite AI and Edge AI. Autonomous vehicles cannot pursue the same data processing capabilities as data centers. Yet, most of the existing deep learning algorithms rely on powerful central processing chips. In addition to supplying energy to the high-performance chip, a large number of cables are required to transmit signals. To go green and low-carbon, we need to improve deep learning algorithms. Light-weight network structures need to be developed so that they can run in low-power chips. At the same time, this low-power chip can be as close to the sensor as possible, reducing the transmission signal latency. This distributed artificial intelligence at the sensor edge is called edge AI [Shi et al., 2020b], and will be a focus of the future research.

Bibliography

[Batouli et al., 2020] Batouli, G., Guo, M., Janson, B., and Marshall, W. (2020). Analysis of pedestrian-vehicle crash injury severity factors in colorado 2006–2016. Accident Analysis & Prevention, 148:105782.

[Bosquet et al., 2020] Bosquet, B., Mucientes, M., and Brea, V. M. (2020). Stdnet: Exploiting high resolution feature maps for small object detection. Engineering Applications of Artificial Intelligence, 91:103615.

[Brunetti et al., 2018] Brunetti, A., Buongiorno, D., Trotta, G. F., and Bevilacqua, V. (2018). Computer vision and deep learning techniques for pedestrian detection and tracking: A survey. Neurocomputing, 300:17–33.

[Caesar et al., 2020] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. (2020). nuscenes: A multimodal dataset for autonomous driving. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 11621–11631.

[Calvi et al., 2020] Calvi, A., D'Amico, F., Ferrante, C., and Ciampoli, L. B. (2020). Effectiveness of augmented reality warnings on driving behaviour whilst approaching pedestrian crossings: A driving simulator study. Accident Analysis & Prevention, 147:105760.

[Chen et al., 2017] Chen, X., Ma, H., Wan, J., Li, B., and Xia, T. (2017). Multi-view 3d object detection network for autonomous driving. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 1907–1915.

[Cheng et al., 2018] Cheng, J., Wang, P.-s., Li, G., Hu, Q.-h., and Lu, H.-q. (2018). Recent advances in efficient computation of deep convolutional neural networks. Frontiers of Information Technology & Electronic Engineering, 19(1):64–77.

[Cui et al., 2021] Cui, Y., Chen, R., Chu, W., Chen, L., Tian, D., Li, Y., and Cao, D. (2021). Deep learning for image and point cloud fusion in autonomous driving: A review. IEEE Transactions on Intelligent Transportation Systems.

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), volume 1, pages 886–893. Ieee.

[Dollár et al., 2009] Dollár, P., Wojek, C., Schiele, B., and Perona, P. (2009). Pedestrian detection: A benchmark. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 304–311. IEEE.

[Duan et al., 2019] Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., and Tian, Q. (2019). Centernet: Keypoint triplets for object detection. In Proceedings of the IEEE/CVF international conference on computer vision, pages 6569–6578.

[Everingham et al., 2010] Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. International journal of computer vision, 88(2):303–338.

[Feng et al., 2020] Feng, D., Haase-Schütz, C., Rosenbaum, L., Hertlein, H., Glaeser, C., Timm, F., Wiesbeck, W., and Dietmayer, K. (2020). Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. IEEE Transactions on Intelligent Transportation Systems, 22(3):1341–1360.

[Fukushima and Miyake, 1982] Fukushima, K. and Miyake, S. (1982). Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In Competition and cooperation in neural nets, pages 267–285. Springer.

[Geiger et al., 2013] Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. The International Journal of Robotics Research, 32(11):1231–1237.

[Geiger et al., 2012] Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In 2012 IEEE Conference on Computer Vision and Pattern Recognition, pages 3354–3361. IEEE.

[Geyer et al., 2020] Geyer, J., Kassahun, Y., Mahmudi, M., Ricou, X., Durgesh, R., Chung, A. S., Hauswald, L., Pham, V. H., Mühlegg, M., Dorn, S., et al. (2020). A2d2: Audi autonomous driving dataset. arXiv preprint arXiv:2004.06320.

[Girshick, 2015] Girshick, R. (2015). Fast r-cnn. In Proceedings of the IEEE international conference on computer vision, pages 1440–1448.

[Girshick et al., 2014] Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 580–587.

[Girshick et al., 2015] Girshick, R., Iandola, F., Darrell, T., and Malik, J. (2015). Deformable part models are convolutional neural networks. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 437–446.

[Goverment, 2020] Goverment, F. (2020). Limitations de vitesse.

[Guo et al., 2016] Guo, Y., Yao, A., and Chen, Y. (2016). Dynamic network surgery for efficient dnns. In Proceedings of the 30th International Conference on Neural Information Processing Systems, page 1387–1395.

[Han et al., 2011] Han, J., Pei, J., and Kamber, M. (2011). Data mining: concepts and techniques. Elsevier.

[Han et al., 2020] Han, M., Wang, Y., Chang, X., and Qiao, Y. (2020). Mining inter-video proposal relations for video object detection. In European Conference on Computer Vision, pages 431–446. Springer.

[He et al., 2015] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence, 37(9):1904–1916.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778.

[Hearst et al., 1998] Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., and Scholkopf, B. (1998). Support vector machines. IEEE Intelligent Systems and their applications, 13(4):18–28.

[Huang et al., 2018] Huang, X., Cheng, X., Geng, Q., Cao, B., Zhou, D., Wang, P., Lin, Y., and Yang, R. (2018). The apolloscape dataset for autonomous driving. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 954–960.

[Hur and Roth, 2020] Hur, J. and Roth, S. (2020). Self-supervised monocular scene flow estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7396–7405.

[Krizhevsky et al., 2017] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. Communications of the ACM, 60(6):84–90.

[Lang et al., 2019] Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., and Beijbom, O. (2019). Pointpillars: Fast encoders for object detection from point clouds. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 12697–12705.

[Law and Deng, 2018] Law, H. and Deng, J. (2018). Cornernet: Detecting objects as paired keypoints. In Proceedings of the European conference on computer vision (ECCV), pages 734–750.

[Lazebnik et al., 2006] Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In 2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06), volume 2, pages 2169–2178. IEEE.

[LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324.

[Li et al., 2019a] Li, B., Ouyang, W., Sheng, L., Zeng, X., and Wang, X. (2019a). Gs3d: An efficient 3d object detection framework for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1019–1028.

[Li et al., 2019b] Li, P., Chen, X., and Shen, S. (2019b). Stereo r-cnn based 3d object detection for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7644–7652.

[Li et al., 2020] Li, P., Zhao, H., Liu, P., and Cao, F. (2020). Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving. In European Conference on Computer Vision, pages 644–660. Springer.

[Lin et al., 2017a] Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017a). Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2117–2125.

[Lin et al., 2017b] Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017b). Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision, pages 2980–2988.

[Lin et al., 2014] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In European conference on computer vision, pages 740–755. Springer.

[Liu et al., 2016] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In European conference on computer vision, pages 21–37. Springer.

[Liu et al., 2022] Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D., and Han, S. (2022). Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. arXiv preprint arXiv:2205.13542.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. International journal of computer vision, 60(2):91–110.

[Lung et al., 2021] Lung, K.-Y., Chang, C.-R., Weng, S.-E., Lin, H.-S., Shuai, H.-H., and Cheng, W.-H. (2021). Rosnet: Robust one-stage network for ct lesion detection. Pattern Recognition Letters, 144:82–88.

[Ma et al., 2020] Ma, X., Liu, S., Xia, Z., Zhang, H., Zeng, X., and Ouyang, W. (2020). Rethinking pseudo-lidar representation. In European Conference on Computer Vision, pages 311–327. Springer.

[McCulloch and Pitts, 1943] McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4):115–133.

[Ng and Henikoff, 2003] Ng, P. C. and Henikoff, S. (2003). Sift: Predicting amino acid changes that affect protein function. Nucleic acids research, 31(13):3812–3814.

[Pang et al., 2020] Pang, S., Morris, D., and Radha, H. (2020). Clocs: Camera-lidar object candidates fusion for 3d object detection. In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 10386–10393. IEEE.

[Peng et al., 2021] Peng, L., Liu, F., Yu, Z., Yan, S., Deng, D., Yang, Z., Liu, H., and Cai, D. (2021). Lidar point cloud guided monocular 3d object detection. arXiv preprint arXiv:2104.09035.

[Qi et al., 2018] Qi, C. R., Liu, W., Wu, C., Su, H., and Guibas, L. J. (2018). Frustum pointnets for 3d object detection from rgb-d data. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 918–927.

[Qi et al., 2017a] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017a). Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 652–660.

[Qi et al., 2017b] Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017b). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems, 30.

[Rasouli et al., 2017] Rasouli, A., Kotseruba, I., and Tsotsos, J. K. (2017). Agreeing to cross: How drivers and pedestrians communicate. In IEEE Intelligent Vehicles Symposium (IV), pages 264–269.

[Redmon et al., 2016] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 779–788.

[Redmon and Farhadi, 2017] Redmon, J. and Farhadi, A. (2017). Yolo9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7263–7271.

[Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems, pages 91–99.

[Rosenblatt, 1958] Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, 65(6):386.

[Sabri and Fauza, 2018] Sabri, M. and Fauza, A. (2018). Analysis of vehicle braking behaviour and distance stopping. IOP Conference Series: Materials Science and Engineering, 309:012020.

[Shi et al., 2019] Shi, S., Wang, X., and Li, H. (2019). Pointrcnn: 3d object proposal generation and detection from point cloud. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 770–779.

[Shi et al., 2020a] Shi, S., Wang, Z., Shi, J., Wang, X., and Li, H. (2020a). From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. IEEE transactions on pattern analysis and machine intelligence.

[Shi et al., 2020b] Shi, Y., Yang, K., Jiang, T., Zhang, J., and Letaief, K. B. (2020b). Communication-efficient edge ai: Algorithms and systems. IEEE Communications Surveys & Tutorials, 22(4):2167–2191.

[Simonelli et al., 2019] Simonelli, A., Bulo, S. R., Porzi, L., López-Antequera, M., and Kontschieder, P. (2019). Disentangling monocular 3d object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1991–1999.

[Simonyan and Zisserman, 2014] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[Sindagi et al., 2019] Sindagi, V. A., Zhou, Y., and Tuzel, O. (2019). Mvx-net: Multimodal voxelnet for 3d object detection. In 2019 International Conference on Robotics and Automation (ICRA), pages 7276–7282. IEEE.

[Sun et al., 2020] Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al. (2020). Scalability in perception for autonomous driving: Waymo open dataset. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 2446–2454.

[Szegedy et al., 2015] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1–9.

[Tavanaei et al., 2019] Tavanaei, A., Ghodrati, M., Kheradpisheh, S. R., Masquelier, T., and Maida, A. (2019). Deep learning in spiking neural networks. Neural networks, 111:47–63.

[Uijlings et al., 2013] Uijlings, J. R., Van De Sande, K. E., Gevers, T., and Smeulders, A. W. (2013). Selective search for object recognition. International journal of computer vision, 104(2):154–171.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

[Viale et al., 2021] Viale, A., Marchisio, A., Martina, M., Masera, G., and Shafique, M. (2021). Carsnn: An efficient spiking neural network for event-based autonomous cars on the loihi neuromorphic research processor. In 2021 International Joint Conference on Neural Networks (IJCNN), pages 1–10. IEEE.

[Visscher, 2008] Visscher, P. M. (2008). Sizing up human height variation. Nature genetics, 40(5):489–490.

[Wang et al., 2009] Wang, X., Han, T. X., and Yan, S. (2009). An hog-lbp human detector with partial occlusion handling. In 2009 IEEE 12th international conference on computer vision, pages 32–39. IEEE.

[Wang et al., 2019] Wang, Y., Chao, W.-L., Garg, D., Hariharan, B., Campbell, M., and Weinberger, K. Q. (2019). Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 8445–8453.

[Wu et al., 2016] Wu, J., Leng, C., Wang, Y., Hu, Q., and Cheng, J. (2016). Quantized convolutional neural networks for mobile devices. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4820–4828.

[Yan et al., 2023] Yan, J., Liu, Y., Sun, J., Jia, F., Li, S., Wang, T., and Zhang, X. (2023). Cross modal transformer: Towards fast and robust 3d object detection. arXiv preprint arXiv:2301.01283.

[Yan et al., 2018] Yan, Y., Mao, Y., and Li, B. (2018). Second: Sparsely embedded convolutional detection. Sensors, 18(10):3337.

[Yim et al., 2017] Yim, J., Joo, D., Bae, J., and Kim, J. (2017). A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4133–4141.

[Zeiler and Fergus, 2014] Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In European conference on computer vision, pages 818–833. Springer.

[Zhang et al., 2021a] Zhang, H., Pop, D. O., Rogozan, A., and Bensrhair, A. (2021a). Accelerate high resolution image pedestrian detection with non-pedestrian area estimation. IEEE Access, 9:8625–8636.

[Zhang et al., 2021b] Zhang, H., Rogozan, A., and Bensrhair, A. (2021b). Cross-modal verification for 3d object detection. In 29th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2021, pages 195–200. ESANN.

[Zhang et al., 2021c] Zhang, H., Rogozan, A., and Bensrhair, A. (2021c). Cross-modal verification for 3d object detection. In ESANN.

[Zhang et al., 2022] Zhang, H., Rogozan, A., and Bensrhair, A. (2022). An enhanced n-point interpolation method to eliminate average precision distortion. Pattern Recognition Letters, 158:111–116.

[Zhang et al., 2016] Zhang, L., Lin, L., Liang, X., and He, K. (2016). Is faster r-cnn doing well for pedestrian detection? In European conference on computer vision, pages 443–457. Springer.

[Zhang et al., 2015] Zhang, X., Zou, J., He, K., and Sun, J. (2015). Accelerating very deep convolutional networks for classification and detection. IEEE transactions on pattern analysis and machine intelligence, 38(10):1943–1955.

[Zhou and Tuzel, 2018] Zhou, Y. and Tuzel, O. (2018). Voxelnet: End-to-end learning for point cloud based 3d object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4490–4499.

[Zhu, 2004] Zhu, M. (2004). Recall, precision and average precision. Department of Statistics and Actuarial Science, University of Waterloo, Waterloo, 2:30.