TUM School of Computation, Information and Technology
Technical University of Munich

Interdisciplinary Project

LiDAR 3D Object Detection for Roadside Infrastructure Sensors Using Transformer

LiDAR 3D Objekt Detektion für Verkehrsinfrastruktursensoren mit Transformer

Supervisor Prof. Dr.-Ing. habil. Alois C. Knoll

Advisor Walter Zimmer, M.Sc.

Author Huu Tung Nguyen

Date August 31, 2023 in Munich

Disclaimer

I confirm that this interdisciplinary project is my own work and I have documented all sources and material used.

Munich, August 31, 2023

(Huu Tung Nguyen)

Abstract

In the pursuit of advancing autonomous driving, 3D object detection stands as a crucial task. Traditional Light and Ranging Sensors (LiDARs), predominantly positioned atop vehicles, struggle with a restricted field of view and frequent occlusions. Recognizing this limitation, the focus has shifted towards leveraging infrastructure LiDARs strategically placed at elevated positions like light poles or gantry bridges to counteract these issues and enhance safety. Our research deepens into this context, seamlessly integrating the TUMTraf dataset with the OpenPCDet codebase. The core of our study is the PointPillars model and its more advanced counterpart, CT3D, which extends the PointPillars model with the attention mechanism. We meticulously compare these models, scrutinizing their performance and inference speeds. In our study, we introduced a post-processing overlap filter to further refine detection accuracy, aiming to remove overlapping detections. The results underscored the prowess of CT3D, but a detailed exploration into inference time illuminated PointPillars as the more time-efficient model, making it a contender for real-time deployments. The prospective integration of TensorRT can reduce its inference time even further.

Zusammenfassung

Der Fortschritt im Bereich autonomes Fahren ist ohne die 3D-Objekterkennung kaum vorstellbar. Herkömmliche auf englisch *Light and Ranging Sensors* (LiDARs), die überwiegend auf Fahrzeugen angebracht sind, haben mit einem eingeschränkten Sichtfeld und häufigen Verdeckungen zu kämpfen. Angesichts dieser Einschränkungen hat sich der Schwerpunkt auf die Nutzung von LiDARs für die Infrastruktur verlagert, die strategisch an erhöhten Positionen wie Lichtmasten oder Schilderbrücken platziert werden, um diesen Problemen entgegenzuwirken und die Sicherheit der Verkehrteilnehmer zu erhöhen. Unsere Forschung beschäftigt sich mit diesem Thema und integriert den TUMTraf-Datensatz in die OpenPCDet Codebasis. Der Essenz unserer Studie sind das PointPillars-Modell und sein fortschrittlicheres Gegenstück CT3D, dass das PointPillars-Modell um die Transformer Architektur erweitert. Wir führen detailierte Vergleiche zwischen diesen Modellen durch und untersuchen ihre Performance und Inferenzgeschwindigkeit. Um die Erkennungsgenauigkeit weiter zu verbessern, haben wir in unserer Studie einen Überlappungsfilter eingeführt, um überlappende Erkennungen zu entfernen. Die Ergebnisse bestätigen die Performance von CT3D, aber eine detaillierte Untersuchung der Inferenzzeit ergab, dass PointPillars das zeiteffizientere Modell ist, was es zu einem Anwärter für Echtzeiteinsätze macht. Die künftige Integration von TensorRT bietet das Potenzial, die Inferenzzeit noch weiter zu reduzieren.

Contents

Chapter 1

Introduction

Autonomous driving technology has experienced significant advancement in recent years, laying the foundation for the future of transportation. Central to these advancements is the capability for a vehicle to perceive its environment accurately, with 3D object detection playing a fundamental role. The motivation for this study arises from the understanding that achieving robust 3D object detection is essential for the safe and efficient operation of autonomous vehicles. In our study, we primarily focus on the modality of Light Detection and Ranging (LiDAR), which serves as a cornerstone for the perception of the car. It represents its environment as a point cloud, a collection of 3D points that, compared to traditional images, offers the critical advantage of depth information. This depth enables a comprehensive understanding of the surroundings. Nevertheless, the inherent unordered nature of point clouds introduces challenges. Point cloud processing necessitates meticulous pre-processing to organize points and extract point features, making direct data utilization complex. While a multitude of methods like grid-based and point-based techniques exist to process these point clouds [Shi+21], there is always room for enhancement, steering the direction of our research questions:

1. How do infrastructure datasets, with elevated sensor positioning, improve 3D object detection compared to traditional datasets?
2. How do traditional 3D object detectors perform on infrastructure datasets?
3. Can the integration of Transformer architecture with 3D object detection models offer superior performance in terms of accuracy and efficiency?
4. What are the specific advantages of using the TUM Traffic Dataset (TUMTraf) for infrastructure-based 3D object detection?

Existing 3D object detection methodologies predominantly detect objects from the ego vehicle's viewpoint [Wu+22]. Although robust, these strategies struggle with occlusions and restricted fields of view, posing potential safety hazards. Infrastructure datasets, however, present a promising alternative. With sensors on infrastructures like light posts and gantry bridges, the resulting scans cover a larger area with a broader field of view, diminishing occlusion and leading the way for our research goals. This research aims to underscore the advantages of infrastructure datasets, particularly leveraging the TUMTraf [Zim+23] dataset. The PointPillars model [Lan+19] is employed as a benchmark for 3D object detection. Still, our study focuses on the effect of deploying the Transformer architecture [Vas+17] using CT3D [She+21] as a comparison model. The expected results hinge on the premise that Transformers, renowned for their attention mechanism and pivotal role in deep learning, especially in natural language processing, will enhance the 3D object detection performances. Our contributions are three-fold:

1. Meticulous evaluation of the PointPillar model on the TUM Traffic dataset.
2. A focused ablation study on integrating Transformer architecture with 3D object detection using infrastructure LiDAR with the CT3D model.
3. An ablation study scrutinizes various filtering method effects.

The trajectory of this research unfolds over a year as shown in fig. 1.1, segmented into methodical phases: an initial literature review, the development of a preliminary functional prototype to gauge the performance of 3D object detection, establishing an initial baseline, the integration of Transformers for enhancing 3D object detection, a comprehensive evaluation of both methodologies and, ultimately, refinement and optimization. Subsequent chapters give a basic understanding of the area of 3D object detection. They provide in-depth knowledge, examine the architectural distinctions of the models employed, outline our experimental setup, and present detailed evaluation results, leveraging various performance metrics. This thorough exploration underscores our methodology and the improvements our approach introduces to 3D object detection.

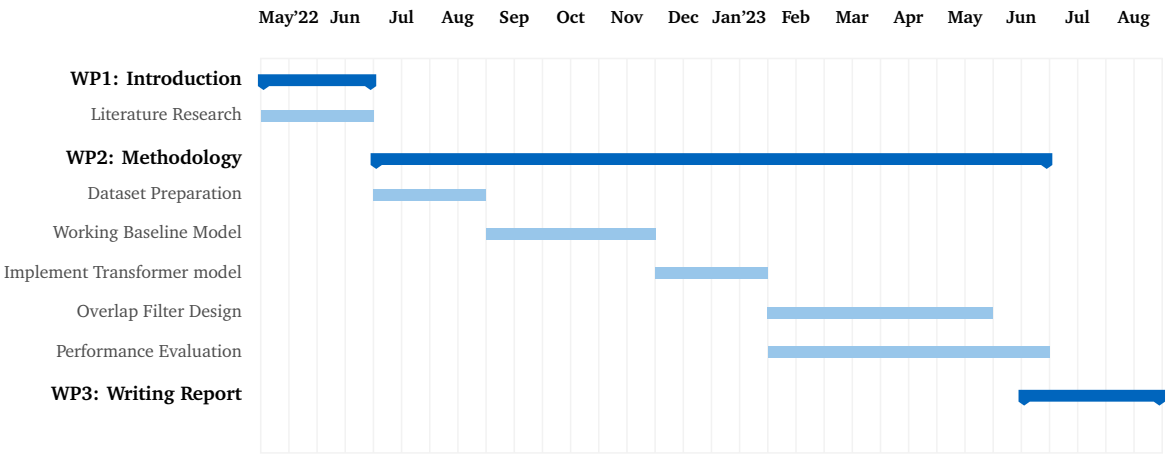


Figure 1.1: Project Time Schedule from May 2022 to August 2023

Chapter 2

Related Work

2.1 Point Cloud Representation

In the contemporary scientific landscape, point clouds have gained considerable momentum across various disciplines, such as computer vision, robotics, autonomous driving, and virtual reality. At the same time, point clouds provide an invaluable three-dimensional representation of the real world but pose substantial challenges due to their large data size, inherent noise, and partial coverage. Given point clouds' irregular, unstructured, and unordered nature, direct processing is often impractical. Therefore, the need to devise efficient and effective methodologies for extracting valuable information from point clouds is crucial. This section provides an overview of prevalent approaches employed in this domain.

2.1.1 Voxel-based

A straightforward yet practical approach entails discretizing the point cloud into a grid structure called voxels. This voxelization procedure renders the data compatible with standard methodologies, such as two-dimensional convolutional neural networks (2D CNNs). Moreover, the process can be extended to three dimensions by implementing three-dimensional CNNs or sparse convolutions, as suggested in [Shi+21]. Due to their speed and efficiency, Voxel-based methods are frequently employed for proposal networks in bird's eye view contexts, such as in [Che+17b] and 3D. However, a notable limitation of CNNs is the constrained receptive field, which depends on kernel size.

2.1.2 Point-based

In contrast, point-based approaches pioneered by PointNet [Qi+17a] to directly work with the point cloud without discretization. This approach extracts point cloud features using a Multilayer Perceptron (MLP). Despite its intuitive appeal, a major drawback lies in the size limitation of the point cloud, which significantly impacts performance.

2.1.3 Point-Voxel-based

Bridging the gap between voxel-based and point-based methodologies, models such as Point-Voxel CNN [Liu+19] leverage both strengths. Initially, voxel-based feature aggregation is applied to the input point cloud, followed by direct extraction of point features from the

point cloud. These two distinct feature sets are combined using various techniques, such as attention mechanisms. This hybrid approach strives to balance the strengths and limitations of both voxel-based and point-based methodologies, offering a comprehensive solution to point cloud representation challenges.

2.1.4 Bird's-Eye View Methods

Bird's-eye view (BEV) methods project the 3D point cloud onto a 2D plane, usually the horizontal plane, and apply 2D detection methods on the resulting image. This approach allows for the use of well-established image-based object detection methods, such as the Region-CNN (R-CNN) [Gir+14], Faster-R-CNN[Ren+16], SSD [Liu+16], and YOLO [Red+16]. However, information about the vertical structure of objects may be lost in the projection.

2.2 3D Object Detection

The principal objective of 3D object detection is identifying and localizing objects within a given scene. Typically, this involves processing an input point cloud to discern the spatial positioning of objects and classify their type. This task bears considerable significance across several fields, such as autonomous navigation, robotics, and computer vision, with the overarching aim of creating systems that can accurately interpret and interact with their environment. Recent advances in deep learning have introduced novel techniques and models that significantly enhance the performance of 3D object detection algorithms, as will be explored in later sections. Over the years, several approaches have been proposed for 3D object detection. These can broadly be divided into four categories: voxel-based, point-based, point-voxel-based, and bird's-eye view methods, as discussed in the previous section. As voxel-based methods, VoxelNet [ZT17] naively partitions 3D space into voxels and aggregates point features for each voxel to apply dense 3D convolutions for context generation. SECOND [YML18] improves the efficiency of the 3D convolution operation by introducing sparse 3D convolutions. On the other side, point-based method such as PointRCNN [SWL19] generates proposals with the help of a feature extractor such as PointNet, PointNet++ [Qi+17b], or PointNext [Qia+22]. The generated proposals are 3D bounding boxes consisting of the 3D location (x, y, z), the dimensions (l, w, h), and the rotation around the z-axis. For further refinement, the proposals are subjected to 3D RoI pooling. This step ensures that each proposal, irrespective of its original size or shape, is converted into a fixed-size feature representation. By extracting and pooling the features within these proposed regions, it becomes possible to efficiently process varying bounding box sizes through the subsequent layers of the network. This pooled representation streamlines the computational process and facilitates a more accurate classification and regression for the final 3D object detection task. Despite all efficiency, grid-based methods induce fine-grained information loss due to discretization. Therefore, point-voxel-based methods are introduced, Point-Voxel CNN (PVCNN) [Liu+19] to fuse point and voxel features. A newly introduced model PllarNext [LLY23], which builds upon PointPillars, extends the PointPillars architecture that consists of a Pillar Encoder, a 2D CNN backbone, and a detection head by inserting an ASPP [Che+17a] neck between backbone and detection head. The neck utilizes the aggregated features from the backbone to enlarge the receptive field and fuse multi-scaled context. ASPP is used in semantic segmentation to segment objects with multiple scales. PillarNext uses this property instead to segment objects for context with various scales.

2.3 Transformer

The Transformer model, introduced by Vaswani et al. in [Vas+17], represents a significant advancement in deep learning, particularly for sequence-to-sequence tasks. Predicated on the self-attention mechanism, the Transformer has revolutionized numerous areas, offering impressive results in natural language processing, image recognition, and even 3D object detection. Unlike traditional recurrent or convolutional layers, the self-attention mechanism processes data in order or localized fields. This gives the model a global view of the input and allows it to consider the entire context when generating each element of the output sequence. This mechanism is then used in the multi-head attention architecture, where the input is divided into several parts, each of which is processed by a self-attention mechanism independently. The outputs of these mechanisms are then concatenated and linearly transformed to produce the final result. This approach allows the Transformer model to pay attention to information from different positions simultaneously, enhancing its ability to understand complex patterns and relationships.

In computer vision, Transformer models, such as Vision Transformers (ViT) [Dos+21], have shown that they can be applied directly to image patches, treating them as a sequence, and still achieve competitive performance on image classification tasks. In 3D object detection, Transformers offer the potential for more expressive feature interactions. Traditional convolutional-based models have limitations due to their localized receptive fields, making capturing long-range dependencies within the data difficult. With their global self-attention mechanism, transformer models can potentially overcome this hurdle. Moreover, Transformer models are known for their high interpretability. The attention maps generated during their operation can provide insights into what parts of the input sequence the model focuses on to create each output piece, aiding in understanding the model's decisions. In the subsequent sections, we will explore how integrating Transformer architectures with the PointPillars model can enhance 3D object detection, potentially leading to a more robust and efficient model for infrastructure-based 3D object detection.

Chapter 3

Infrastructure Dataset

Infrastructure datasets present a novel approach to data collection for autonomous systems. Unlike traditional ego-centric datasets, where data is collected from the vehicle's perspective, infrastructure datasets are captured from an elevated view, usually from high positions within urban infrastructure, such as traffic light posts, overpasses, or high-rise buildings. This change of perspective offers a unique set of advantages and new challenges. The advantages of infrastructure datasets are an elevated position offers a broader field of view, enabling the detection of objects farther away from the sensor. This extended detection range offers the autonomous system a longer reaction time, improving overall system safety. In urban environments, lower-placed sensors often suffer from occlusions due to large vehicles, pedestrians, or other urban structures blocking the line of sight. An elevated sensor position, as afforded by infrastructure-based sensing, significantly mitigates this problem. Infrastructure datasets can capture more diverse perspectives of the environment, which can be particularly useful in complex urban scenarios with multiple dynamic actors.

3.1 TUMTraf Dataset

The TUM Traffic Dataset (TUMTraf) is a multi-modal dataset consisting of image and LiDAR data. It also includes highway and intersection data, providing a range of vehicle maneuvers and scenarios. This work mainly focuses on the intersection dataset [Zim+23], which consists of 4.8k camera images and 4.8k LiDAR point cloud frames containing 57.4 labeled 3D boxes.

Several infrastructure datasets have been introduced to facilitate research in this area. One prominent example is the A9 [Zim+23] and the DAIR-V2X [Yu+22] dataset, which are large-scale, real-world datasets designed explicitly for 3D object detection research from infrastructure LiDAR sensors. It provides high-resolution LiDAR scans and corresponding annotations, which will be used in this work to investigate the application of Transformer mechanisms for infrastructure 3D object detection.

3.2 Statistics

A detailed analysis of the TUMTraf dataset reveals a significant distribution of ten distinct object classes, namely: CAR, TRUCK, TRAILER, VAN, PEDESTRIAN, MOTORCYCLE, BUS, BICYCLE, EMERGENCY VEHICLES, and OTHER. Figure 3.1 shows the occurrences of all ten classes in a logarithmic scale for better visual comparison. On the first look, we can observe

that the CAR class is the most dominant one with 22773 bounding boxes, followed by the VAN class. On average, we have 3805 occurrences across all ten categories, reflecting the occurrences for the TRUCK, TRAILER, and PEDESTRIAN classes since they are close to it. The least represented categories are EMERGENCY_VEHICLE and OTHER with 142 and 84 occurrences, respectively. In fig. 3.2 shows the average number of points per class. On

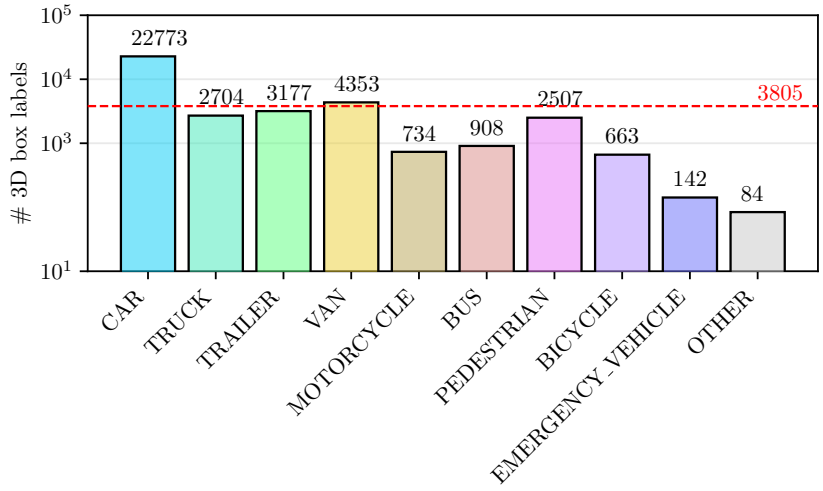


Figure 3.1: Class occurrences [Zim+23].

average, a bounding box contains 103 LiDAR points across all categories. The TRAILER and BUS classes have the highest number of points on average, with 328 and 222, respectively. The high number of points can be attributed to its dimensions. The two classes are the largest in length and height, where the TRUCK has a length of 3.11m and a height of 3.43m, and the TRAILER a length of 10.19m and a height of 3.65m as shown in table 3.1. The classes with the least amount of points are MOTORCYCLE, BICYCLE, and PEDESTRIAN, with magnitudes of 21, 20, and 14. Following our prior reasoning, we can confirm that it is due to the most diminutive dimensions of these classes, where the pedestrian has the very most minor extent.

Table 3.1: The total number of 3D box labels, average dimensions in meters, and the average number of 3D LiDAR points among all classes [Zim+23].

Class	#Labels	⌀Length	⌀Width	⌀Height	⌀Points
Car	**22,773**	4.27	1.91	1.59	34.03
Truck	2,704	3.11	2.90	3.43	116.87
Trailer	3,177	10.19	**3.12**	**3.65**	**328.36**
Van	4,353	6.35	2.52	2.47	86.11
Motorcycle	734	1.90	0.83	1.60	21.23
Bus	908	**12.65**	2.95	3.27	222.36
Pedestrian	2,507	0.80	0.73	1.72	14.98
Bicycle	663	1.57	0.74	1.72	20.95
Emergency Vehicle	142	6.72	2.35	2.35	58.95
Other	84	5.28	1.92	1.90	128.17
Total	38,045	-	-	-	103.20

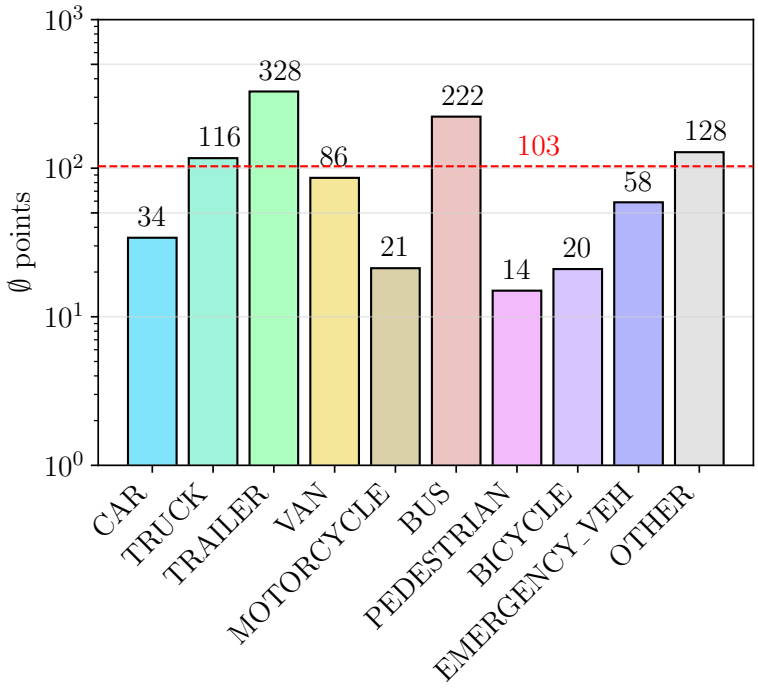


Figure 3.2: Average number of points per class [Zim+23].

3.3 Data Split

In 3D object detection, partitioning our dataset is of essential importance. Therefore, to bolster the reliability and robustness of our 3D object detection model, we ensured a meticulous partitioning of our dataset. We have divided the data into training, validation, and test sets. The split ratio is 80% training, 10% validation, and 10% test set. The dataset is segmented into four subsets, each representing recordings from a separate day and under varied atmospheric conditions. These subsets labeled S1 to S4, encompass continuous camera footage coupled with labeled LiDAR captures. S1 and S2 have a duration of 30 seconds, showcasing scenarios during dusk. S3 offers a more extended glimpse with a 120-second sequence captured under bright daylight and clear skies. Meanwhile, S4 provides a 30-second recording taken during nighttime amidst heavy rainfall. This division inherently incorporates varying environmental factors, ensuring our model remains invariant to daily fluctuations and diverse weather scenarios. Such an approach helps make the model more adaptable and capable of handling real-world inconsistencies. For the actual partitioning, we employed stratified sampling, using object classes as the stratification criterion. Stratified sampling is grounded in maintaining an approximately equal distribution of each class across the subsets. Doing so ensures that no specific class is overrepresented or underrepresented in any given subset. Such an equal distribution across different days and weather conditions aids in constructing a training set that offers universal exposure to the model. Similarly, it ensures the validation and test sets are comprehensive and have a similar object class distribution. The chosen balanced stratification mitigates the potential risk of model overfitting to certain classes during the training. In essence, it helps develop a 3D object detection model that is robust in terms of performance across varying scenarios and generalized, ensuring consistent detection capabilities across diverse object classes and environmental conditions.

Chapter 4

Methodology

This chapter presents our systematic approach, spanning dataset integration, model implementation, inference, and post-processing steps. We start by detailing the integration of the TUMTraf dataset into the OpenPCDet [Tea20] codebase and the subsequent inclusion of the CT3D model. Through detailed adjustments, we have tuned the configurations to be suitable for both training and test pipelines. Our primary focus is on the 3D object detection model, emphasizing the PointPillar model and its enhanced variant, the CT3D model. The latter augments PointPillar by introducing an attention mechanism. Furthermore, we will explore the subtle differences between attention mechanisms, such as self-attention, cross-attention, and tunnel-attention. Moving on to the inference stage, we discuss how our trained model processes unseen data and the means used to generate predictions. This is followed by the post-processing steps, where we discuss the filtering strategies implemented to refine and enhance the raw predictions, ensuring optimal results.

4.1 Integration of TUMTraf Dataset with OpenPCDet Codebase

OpenPCDet is a well-established codebase designed explicitly for point cloud-based object detection tasks. Powered by PyTorch, OpenPCDet has been widely accepted by researchers and developers due to its modular architecture and the ability to support various state-of-the-art point cloud detection networks. It provides a holistic ecosystem for point cloud data preprocessing, complex model architectures, and vital post-processing utilities. Our choice of OpenPCDet as the primary codebase was motivated by several factors:

- **Modularity:** The clear separation of various components in OpenPCDet allows for seamless integration of custom datasets and models.
- **Community Support:** A vast community of researchers and developers supports OpenPCDet. This ensures quick troubleshooting and updates in line with the latest advancements in point cloud detection. Additionally, many state-of-the-art models are also built upon the OpenPCDet codebase.
- **Pipeline:** OpenPCDet offers a whole training pipeline with augmentation, distributed training support, and an inference template.

Integrating the TUMTraf dataset into the OpenPCDet framework required several systematic steps:

1. **Data Format Conversion:** The TUMTraf dataset was initially transformed into a format compatible with OpenPCDet's data loaders. Since the TUMTraf point cloud data is in

`.pcd` format and OpenPCDet expects `.bin` for custom datasets, we have to create a converter from `.pcd` to `.bin` following a `float32` format. Similarly, the TUMTraf labels are naturally in the standard OpenLabel format, but the labels in OpenPCDet are expected to be in KITTI format. Therefore, we create a script that does this conversion from OpenLabel to KITTI format. After converting, we make a serialized dictionary using the OpenPCDet pickle file script. This step ensures the codebase can efficiently ingest the data without redundant transformations during runtime.

2. **Adjusting Configuration Files:** OpenPCDet operates based on configuration files that dictate various parameters for preprocessing, model architecture, and post-processing. These configurations were fine-tuned to account for the specifics of the TUMTraf dataset. We limited the point cloud range from $[0, 64]$ in x direction, $[-64, 64]$ along y-axis and $[2, -8]$ in z-axis, due to the elevated height of the LiDAR sensors. Additionally, we extend the classes with the calculated dimensions discussed in the previous chapter.

3. **Model Architecture Customization:** While OpenPCDet supports numerous models, slight modifications were needed to optimize performance on the TUMTraf dataset, ensuring that the model was tuned to the unique characteristics of the dataset. In our case, we adapted the voxel and pillar grid sizes.

4. **Validation and Testing:** Post-integration, rigorous validation and testing cycles were performed to ascertain that the integration was seamless and devoid of any potential glitches. One such measure is the qualitative analysis of the predicted bounding boxes and whether they align with the ground truth. Another one is if the filtering appropriately worked, but compare the results with and without filtering.

This integration process allowed us to leverage the capabilities of OpenPCDet and ensured that our models were tailor-made for the nuances and intricacies of the TUMTraf dataset.

4.2 PointPillars

PointPillars is a popular model for 3D object detection from point clouds, primarily because of its effective architecture that transforms point clouds into pseudo-images for further processing, its computational efficiency, and its high performance. The PointPillars model comprises three core components: the feature encoder network, the 2D convolutional backbone, and the detection head. The first module, the feature encoder network, is responsible for the processing of the raw point cloud data. Point cloud data is a set of data points in a three-dimensional coordinate system. These data points represent the external surfaces of objects sensed by the LiDAR. However, point clouds are unordered and irregular, making them challenging to process directly. To tackle this issue, the feature encoder network of the PointPillars model transforms the point cloud into a more manageable format known as a pseudo-image. The pseudo-image created by the feature encoder network is a sparse 2D representation of the original 3D point cloud. This transformation retains the key spatial and depth information from the point cloud but presents it in a format more suitable for processing by standard 2D convolutional neural networks (CNNs). The pseudo-image is then passed to the second module in the PointPillars model, the 2D convolutional backbone. This component of the model leverages the strength of 2D CNNs to extract features from the pseudo-image. These features capture valuable information about the spatial relationships and patterns in the data crucial for object detection. Finally, the extracted features are fed into the third module, the detection head. The detection head is a specialized network that performs the actual task

of object detection. It uses the features provided by the 2D convolutional backbone to predict the 3D bounding boxes for the objects in the scene, effectively identifying each detected object's location, size, and orientation.

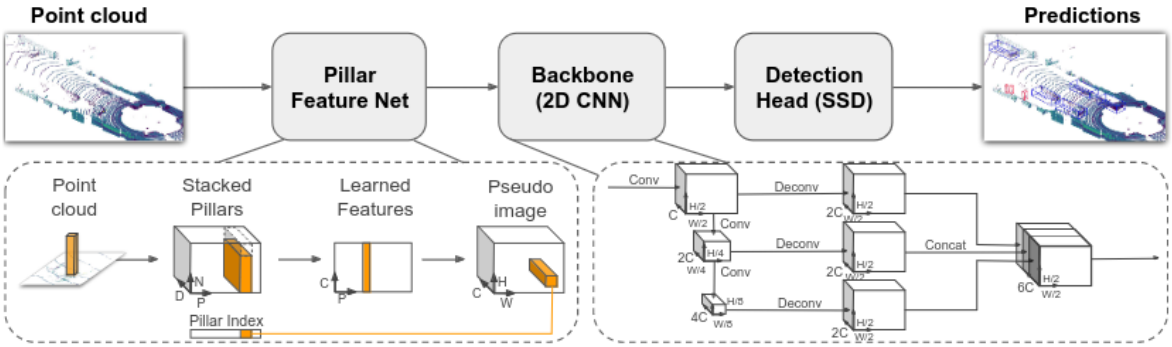


Figure 4.1: PointPillars model illustration: Shows the feature transformation into a pseudo-image which is processed by a 2D CNN backbone.

4.3 CT3D

In the following we discuss different attention mechanism, their differences and how the CT3D module extends PointPillars with channel-wise attention.

4.3.1 Attention Mechanism

In deep learning, attention mechanisms are a class of approaches designed to enable models to focus on the most pertinent information during prediction tasks. This powerful technique has demonstrated remarkable results in various fields, particularly in natural language processing, where it has been successfully applied in models like Transformers [Vas+17]. Three prominent types of attention mechanisms can be identified - self-attention, cross-attention, and tunnel-attention, each with unique characteristics and applications. Self-attention, or intra-attention, allows a model to focus on different parts of its input when producing an output. It computes a weighted sum of all features at every position of the input sequence for each output feature, creating an internal representation of the input that highlights the relevant parts. Self-attention excels in tasks involving sequences, where the model needs to correlate elements that are distant from each other in the input sequence. Cross-attention extends the concept of self-attention by allowing one sequence to attend to another. This mechanism is beneficial in tasks where a model must map one sequence to another, and the alignment between the sequences is unknown a priori. It has been particularly successful in machine translation and other sequence-to-sequence prediction tasks. Tunnel-attention, a more recent development, builds on self-attention but incorporates additional spatial dependencies. It achieves this by dynamically aggregating channel-wise features from different spatial positions of the input, which can enhance the model's ability to capture long-range dependencies in the data.

The CT3D module extends PointPillars by introducing a channel-wise attention mechanism. This mechanism, built on the principles of the tunnel-attention approach, seeks to refine the feature representation by dynamically adjusting the relative importance of different channels in the feature map. In essence, CT3D uses attention to identify which channels

in the feature map are most relevant for the detection task. By focusing on these appropriate channels and down-weighting the less relevant ones, the model can generate a more expressive and task-specific feature representation, which can potentially enhance its detection performance.

4.3.2 Approach

The CT3D module extends the PointPillars model by incorporating a channel-wise transformer module that enables it to use global context information to enrich point features. The process involves a sequence of operations, including encoding input features, decoding proposal features, and finally, prediction and regression stages. First, the raw point cloud data and the preliminary 3D bounding box proposals from the PointPillars model are fed into the channel-wise transformer module. This module employs a transformer architecture to effectively aggregate features across different channels, using proposal-aware context information from all parts of the scene. This operation can help the model capture global contextual cues that a standard PointPillars model might miss due to its locally constrained receptive fields. Once the point features have been enriched with this global context information, they are decoded into a proposal feature representation. The decoding process transforms the encoded point features into a format readily used for downstream tasks. Finally, the translated proposal features are passed to a fully connected network. This network performs two crucial functions: confidence prediction and bounding box regression. The confidence prediction stage estimates the likelihood of each bounding box proposal containing an object. In contrast, the bounding box regression refines the preliminary bounding boxes from the PointPillars model, adjusting their size, orientation, and location to better match the actual objects in the scene. By integrating this channel-wise attention mechanism with the PointPillars architecture, the CT3D module creates a more robust and efficient 3D object detection model capable of effectively leveraging global context information for improved detection performance. The performance of this extended model will be empirically evaluated and discussed in the following chapter.

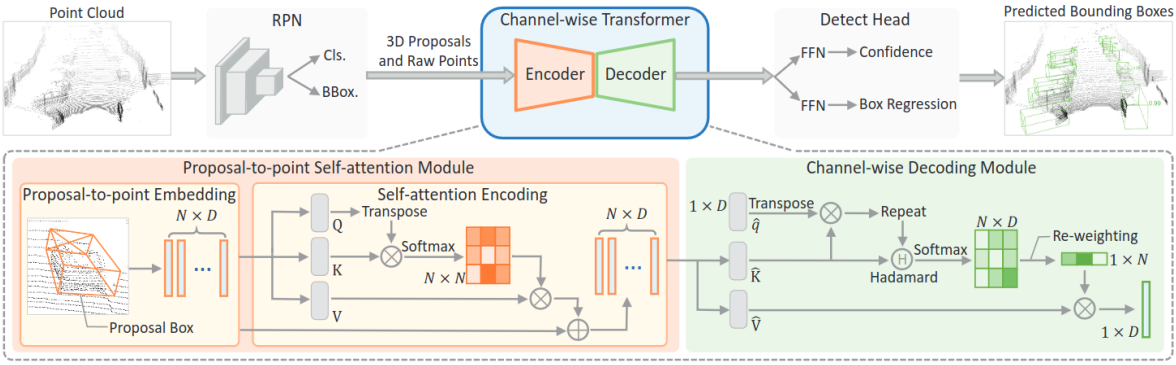


Figure 4.2: PointPillars model illustration: Shows the feature transformation into a pseudo-image which is processed by a 2D CNN backbone.

4.4 Filtering

Post-processing steps can further boost the accuracy and efficiency of the detection results. We will discuss one such approach involving implementing a filtering process based on a

score threshold and overlap detection. The filtering process aims to refine the model's output and remove any potential false positives or redundant detections. This process involves two main stages:

Score Threshold Filtering

The first stage of the filtering process involves setting a threshold for the confidence scores generated by the model. Every detected object is associated with a confidence score that indicates the model's certainty about the detection. Any detections with a confidence score below the specified value are filtered out by setting a score threshold. This reduces the chances of false positives, leading to more reliable and accurate detection results.

Overlap Detection

Following the score threshold filtering, the next stage involves filtering based on overlap. It's not uncommon for a model to produce multiple detections for the same object, particularly in complex scenes. To address this issue, an overlap detection algorithm is applied. We use the Intersection over Union (IoU) calculation algorithm from Meta's PyTorch3D library in our work. The IoU score is a measure of the overlap between two bounding boxes. If the IoU score for two detections exceeds a certain threshold, it indicates that the boxes significantly overlap and likely correspond to the same object. In this case, only the detection with the highest confidence score is kept, while the others are filtered out.

Through this two-stage filtering process, the output of the 3D object detection model is refined and improved, leading to a more precise and reliable set of object detections.

Chapter 5

Evaluation

This research aims not merely to propose approaches to 3D object detection on infrastructure LiDAR but to substantiate their efficacy through rigorous empirical analysis. In this chapter, we present our evaluation of the PointPillars and the CT3D model applied to the TUMTraf dataset. The TUMTraf dataset, as previously outlined, encompasses a rich and varied array of vehicle maneuvers and scenarios.

5.1 Evaluation Metrics

Our evaluation is based on a key metric: the Intersection over Union (IoU). The IoU is a measure of overlap between the ground truth bounding box and the predicted bounding box by the model. It is a popular evaluation metric in object detection tasks due to its straightforward interpretability and robustness. The IoU is calculated as the area of overlap between the predicted bounding box and the ground truth box, divided by the area of the union of the two boxes. A higher IoU score corresponds to a higher degree of overlap, indicating a more accurate prediction.

5.2 Quantitative Results

In our experimental analysis of the test set, we first centered our attention on PointPillars. The baseline configuration was PointPillars without any post-processing, excluding the application of a score threshold filter. Table 5.1 presents the performance metrics of this baseline model. After introducing an overlap filter, which accounted for three specific overlap scenarios in object detection, we noticed a tangible improvement in the model's performance metrics. Notably:

1. The TRUCK and TRAILER classes can overlap since they are typically found together.
2. Overlapping instances of the TRAILER class are permitted since a truck can have multiple trailers.
3. Overlapping instances of pedestrians are also allowed, as they often move in groups, particularly at red lights.

The results of the overlap filter lead to a slight uptick in mIoU scores and an observable increase in precision for nearly all classes. This reaffirms the filter's relevance in aligning the detection process closer to real-world scenarios.

	Baseline			Overlap Filtering		
Class	**Precision**	**Recall**	**mAP**	**Precision**	**Recall**	**mAP**
CAR	79.03	65.84	78.61	79.06	65.73	78.64
TRUCK	**90.09**	80.72	89.89	**90.09**	80.72	89.89
TRAILER	81.52	66.73	81.15	81.11	66.73	80.74
VAN	76.23	57.73	75.75	76.23	57.73	75.75
MOTORCYCLE	85.93	74.88	85.65	85.93	74.88	85.65
BUS	80.39	64.31	80.00	80.39	64.31	80.00
PEDESTRIAN	88.34	84.07	88.10	88.39	84.07	88.16
BICYCLE	84.37	**100.00**	84.05	84.60	**100.00**	84.29
Overall	83.23	74.29	82.90	83.28	74.20	82.94
F1-Score	78.506			78.478		

Table 5.1: Ablation study of PointPillars

Additionally, we conducted several experiments comparing the performance of PointPillars and CT3D, with the baseline being PointPillars without any post-processing except for filtering by a score threshold. Subsequently, we examined the impact of applying an overlap filter and evaluated its effect on the results. Table 5.2 presents the performance metrics of the transformer model compared to PointPillars, which serves as the baseline model. The results indicate that the Transformer model outperforms PointPillars. CT3D's outstanding performance in the BICYCLE class, registering a score of 93.33 compared to PointPillars' 84.05, emerges as a focal point of discussion. On the other hand, CT3D underperformed PointPillars in detecting classes like CAR, TRAILER, VAN, BUS, and PEDESTRIAN. After ap-

	PointPillars			CT3D		
Class	**Precision**	**Recall**	**mAP**	**Precision**	**Recall**	**mAP**
CAR	79.03	65.84	78.61	78.55	64.76	78.12
TRUCK	**90.09**	80.72	89.89	90.10	80.37	89.90
TRAILER	81.52	66.73	81.15	81.51	66.17	81.14
VAN	76.23	57.73	75.75	74.98	57.23	74.48
MOTORCYCLE	85.93	74.88	85.65	87.01	78.25	86.75
BUS	80.39	64.31	80.00	80.36	63.37	79.97
PEDESTRIAN	88.34	84.07	88.10	86.82	84.07	86.56
BICYCLE	84.37	**100.00**	84.05	**98.33**	**100.00**	98.3
Overall	83.23	74.29	82.90	84.66	74.35	**84.35**
F1-Score	78.506			79.170		

Table 5.2: Baseline results of PointPillars and CT3D

plying the overlap filter, we observed slightly improved mIoU scores for both PointPillars and CT3D. Specifically, PointPillars exhibited an increase of 0.05 in mIoU, while CT3D showed an improvement of 0.08 in mIoU. Additionally, we observed a slight increase in precision for almost all classes after the overlap filtering was applied. These findings suggest that the overlap filter contributes to refining the detection results by allowing for specific overlaps that are expected or typical in real-world scenarios. The slight improvement in mIoU and precision across classes indicates the effectiveness of the overlap filter in enhancing the accuracy and quality of the object detection models. In the analysis of detection results with the baseline and the filtering, there is a prominent performance disparity: CT3D excels notably

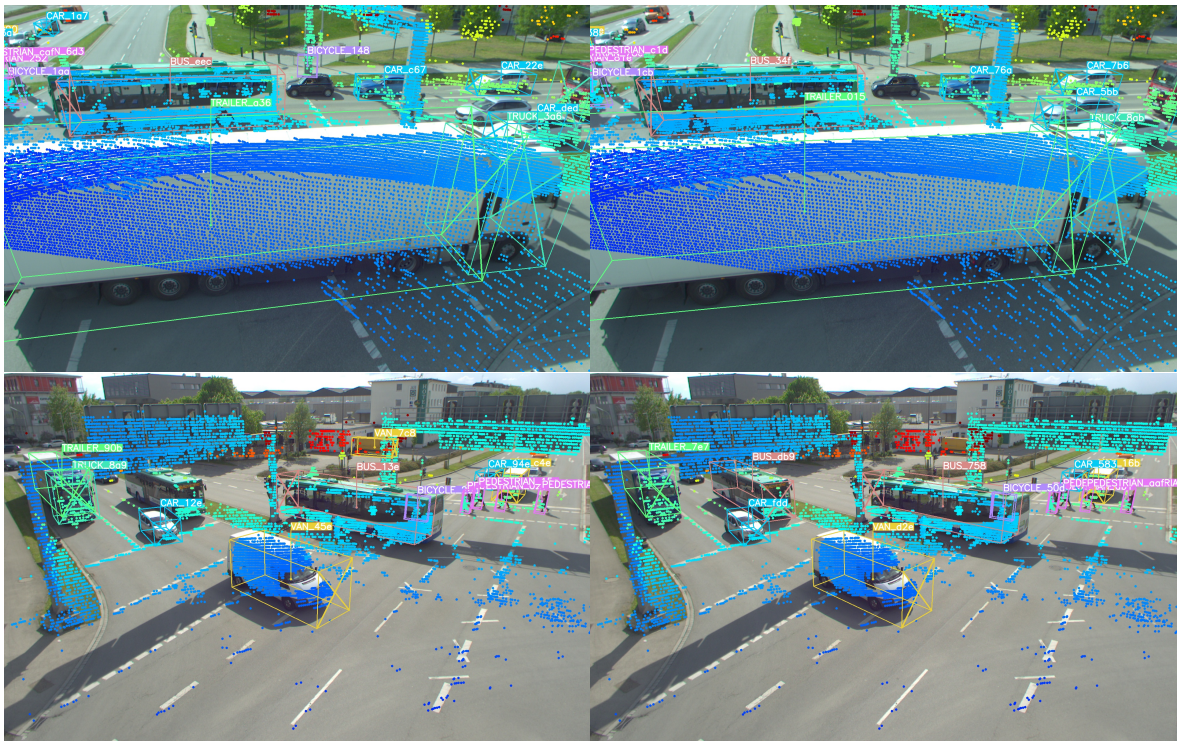
	PointPillars			CT3D		
Class	**Precision**	**Recall**	**mAP**	**Precision**	**Recall**	**mAP**
CAR	79.06	65.73	78.64	78.65	64.64	78.23
TRUCK	**90.09**	80.72	89.89	90.10	80.37	89.90
TRAILER	81.51	66.17	81.14	81.15	66.73	78.82
VAN	76.23	57.73	75.75	75.08	57.06	74.58
MOTORCYCLE	85.93	74.88	85.65	87.11	78.25	86.85
BUS	80.39	64.31	80.00	80.39	63.37	80.00
PEDESTRIAN	88.39	84.07	88.16	87.03	83.67	86.77
BICYCLE	84.60	**100.00**	84.29	**98.33**	**100.00**	98.3
Overall	83.28	74.20	82.94	84.73	74.26	**84.43**
F1-Score	78.478			79.150		

Table 5.3: Results of PointPillars and CT3D with filtering as post-processing step

in detecting bicycles compared to other object categories. One potential reason is the unique structural intricacy of bicycles. Given the transformer architecture of CT3D, it's plausible that the model's self-attention mechanism effectively focuses on distinct features of bicycles, such as its frame or wheels. Contrarily, for more significant object categories such as cars, trailers, vans, and buses, PointPillars demonstrates superior performance. The extensive data produced by these larger objects might be processed more adeptly by PointPillars. While beneficial for bicycles, CT3D's self-attention mechanism may become too excessive in specific scenarios, focusing on localized features and missing more macroscopic details of larger objects.

5.3 Qualitative Results

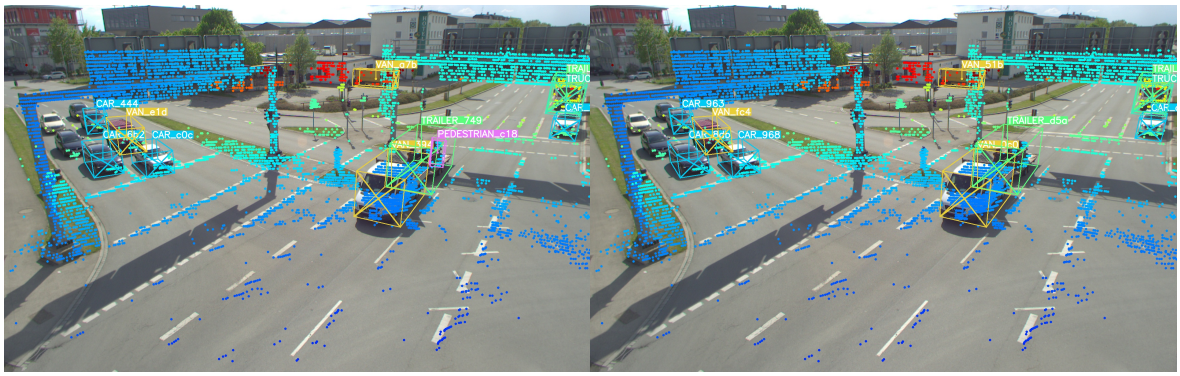
Figure 5.1 presents a qualitative analysis comparing the performance of two object detection models, PointPillars and CT3D. The objective is to assess the differences between the models and evaluate the impact of overlap filtering. The findings indicate that the CT3D model surpasses the PointPillars model regarding object detection capabilities. Specifically, the CT3D model successfully detects and classifies a bus undetected by PointPillars. This outcome highlights the superior accuracy and effectiveness of CT3D in correctly identifying the bus object. Additionally, it is noteworthy that both models successfully detect a bicycle behind the bus despite the occlusion caused by the camera's field of view. This observation showcases the ability of both models to handle challenging scenarios involving partially visible objects. These qualitative results provide valuable insights into the comparative performance of PointPillars and CT3D for object detection. CT3D demonstrates more robust detection capabilities by accurately identifying objects that PointPillars fails to detect. Moreover, both models' successful detection of an occluded bicycle indicates their robustness in handling complex visual scenarios. In the context of the experiment discussed in Figure 5.2, the filter demonstrates the efficacy of an overlap filter in improving the quality of object detection results. The figure presents a comparative analysis before and after the application of the filter. Before filtering, a pedestrian is detected within a trailer, indicating a false positive. However, after applying the overlap filter, the pedestrian detection inside the trailer is successfully removed, leading to a more refined and accurate outcome. This observation suggests that the overlap filter effectively addresses the issue of redundant or erroneous detections by eliminating false positives. The qualitative assessment of the results demonstrates the improved accuracy and



(a) PointPillars (b) CT3D

Figure 5.1: Qualitative results on the test sequence of the baseline model PointPillars and the extended model CT3D. We can see that the CT3D model detects slightly more objects e.g. the bus.

reliability of the object detection when the filter is applied.



(a) Unfiltered (b) Filtered

Figure 5.2: We see a comparison between without filtering overlaps on the left and with on the right. It can be observed that the detected pedestrian inside the trailer is filtered.

5.4 Inference Time Analysis

To gauge the performance efficiency of the two primary models in the study, PointPillars, and CT3D, we evaluated their inference times. These times are critical to understand, especially when considering the potential deployment of these models in real-time systems where

rapid response is paramount and weighing whether accuracy is more important than speed. The benchmarking was conducted on a high-performance NVIDIA RTX 4090 GPU to ensure representative results. The table below illustrates the inference times and frames per second (FPS) for both models, with and without filtering: As observed in table 5.4, PointPillars

	PointPillars	**CT3D**
Without Filtering	16.2 ms/62 FPS	33.2 ms/ 30 FPS
With Filtering	17.5 ms/57 FPS	34.3 ms/ 29 FPS

Table 5.4: Inference time of PointPillars and CT3D on a RTX 4090.

showcases a quicker inference time, achieving up to 62 FPS without filtering and 57 FPS with filtering. Conversely, the CT3D model, incorporating more complex operations such as the self-attention mechanism, displays a slightly lower performance, reaching 30 FPS without filtering and 29 FPS post-filtering. This performance differential underlines the trade-offs between accuracy, as seen in previous sections, and computational efficiency.

Chapter 6

Future Outlook

Our discussion into the area of 3D object detection with the PointPillars and CT3D models has uncovered significant potential, yet there are many points for expansion and improvements:

1. **Pedestrian Detection:** The highest priority is ensuring the safety of traffic participants. Pedestrians are especially vulnerable in the case of traffic accidents. Therefore, enhancing the model's detection capabilities for pedestrians is inevitable. Grid-based models struggle to detect those since reducing the voxel size is always tightly coupled with a loss in inference speed. Additionally, the model might not detect large objects or classify them correctly.

2. **Post-Processing Filter Refinement:** Our current filter, designed to prune false positives, has shown potential. However, the filtering is imperfect, and the rules are pretty simple. We believe refining this filter will better align model outputs with real-world scenarios.

3. **Deployment in Live Systems:** Deploying the model to a live system and testing it on unseen scenarios with real-time performance will offer genuine insights into the model's limitations.

4. **Optimizing Inference Time with TensorRT:** For a model to be deployed on a live system, it has to meet the system requirements and be able to infer in real time. Leveraging TensorRT can potentially reduce the model's inference time.

5. **Transfer Learning.** The DAIR dataset offers a rich repository of learning. By harnessing transfer learning techniques, we can imbibe knowledge from DAIR to boost the performance of our current models.

By addressing these focal points, we anticipate an overall improvement in the performance and robustness of our models, contributing to the advancement of autonomous driving.

Chapter 7

Conclusion

In this report, we conducted experiments comparing the performance of PointPillars and CT3D. Our findings indicate that the transformer-based CT3D model outperforms PointPillars regarding mIoU scores, achieving a higher accuracy of 84.35 compared to PointPillars' 82.90. This demonstrates the superior performance and improved object detection capabilities of the Transformer model. However, it's imperative to highlight an intrinsic trade-off between the models. While CT3D excels in accuracy, PointPillars shines in terms of inference time. This disparity becomes particularly salient when considering real-time or live systems: while CT3D offers refined detection capabilities, its higher latency exceeded our expectations. It might be a hindrance in environments where rapid responses are paramount. Conversely, PointPillars, with its faster inference, might be a preferred choice in scenarios where the inference time is crucial, even at the cost of a slight drop in accuracy. Our innovative addition, to further improve the results, the overlap filter, has further refined detection capabilities. By considering specific exceptions for overlaps in the detection task. The filter allowed overlapping instances of the TRUCK and TRAILER classes, multiple trailers associated with a truck, and overlapping instances of pedestrians moving in groups. Applying this overlap filter resulted in a slight improvement in mood for both PointPillars and CT3D. Furthermore, we observed a slight increase in precision across various classes. These findings highlight the effectiveness of the overlap filter in refining the object detection results. By accounting for expected or common overlaps in real-world scenarios, the filter contributes to improved accuracy and quality of the detection models. Overall, the study demonstrates the advantages of the Transformer model over the baseline model, showcasing its ability to achieve higher mIoU scores and superior object detection performance. Adding the overlap filter further enhances the results, leading to slight improvements in mIoU and precision. The adjacent performance comparison against inference time becomes even more pertinent considering future deployments, especially in live systems. Integrating tools like TensorRT could offer a middle ground, potentially optimizing the models to achieve quicker inference without significant compromises on accuracy. In sum, our study not only underscores the superior detection capabilities of CT3D but also highlights the need to mind the balance between detection accuracy and computational efficiency. Further research and experimentation could focus on refining the overlap filter and exploring other techniques to boost the performance of object detection models.

Bibliography

[Che+17a] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. *DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs*. 2017. arXiv: 1606.00915 [cs.CV].

[Che+17b] Chen, X., Ma, H., Wan, J., Li, B., and Xia, T. *Multi-View 3D Object Detection Network for Autonomous Driving*. 2017. arXiv: 1611.07759 [cs.CV].

[Dos+21] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV].

[Gir+14] Girshick, R., Donahue, J., Darrell, T., and Malik, J. *Rich feature hierarchies for accurate object detection and semantic segmentation*. 2014. arXiv: 1311.2524 [cs.CV].

[Lan+19] Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., and Beijbom, O. *PointPillars: Fast Encoders for Object Detection from Point Clouds*. 2019. arXiv: 1812.05784 [cs.LG].

[LLY23] Li, J., Luo, C., and Yang, X. *PillarNeXt: Rethinking Network Designs for 3D Object Detection in LiDAR Point Clouds*. 2023. arXiv: 2305.04925 [cs.CV].

[Liu+16] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. "SSD: Single Shot MultiBox Detector". In: *Computer Vision – ECCV 2016*. Springer International Publishing, 2016, pp. 21–37. DOI: 10.1007/978-3-319-46448-0_2. URL: https://doi.org/10.1007%2F978-3-319-46448-0_2.

[Liu+19] Liu, Z., Tang, H., Lin, Y., and Han, S. *Point-Voxel CNN for Efficient 3D Deep Learning*. 2019. arXiv: 1907.03739 [cs.CV].

[Qi+17a] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation*. 2017. arXiv: 1612.00593 [cs.CV].

[Qi+17b] Qi, C. R., Yi, L., Su, H., and Guibas, L. J. *PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space*. 2017. arXiv: 1706.02413 [cs.CV].

[Qia+22] Qian, G., Li, Y., Peng, H., Mai, J., Hammoud, H. A. A. K., Elhoseiny, M., and Ghanem, B. *PointNeXt: Revisiting PointNet++ with Improved Training and Scaling Strategies*. 2022. arXiv: 2206.04670 [cs.CV].

[Red+16] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. arXiv: 1506.02640 [cs.CV].

[Ren+16] Ren, S., He, K., Girshick, R., and Sun, J. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. 2016. arXiv: 1506.01497 [cs.CV].

[She+21] Sheng, H., Cai, S., Liu, Y., Deng, B., Huang, J., Hua, X.-S., and Zhao, M.-J. *Improving 3D Object Detection with Channel-wise Transformer*. 2021. arXiv: 2108.10723 [cs.CV].

[Shi+21] Shi, S., Guo, C., Jiang, L., Wang, Z., Shi, J., Wang, X., and Li, H. *PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection*. 2021. arXiv: 1912.13192 [cs.CV].

[SWL19] Shi, S., Wang, X., and Li, H. *PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud*. 2019. arXiv: 1812.04244 [cs.CV].

[Tea20] Team, O. D. *OpenPCDet: An Open-source Toolbox for 3D Object Detection from Point Clouds*. https://github.com/open-mmlab/OpenPCDet. 2020.

[Vas+17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. *Attention Is All You Need*. 2017. arXiv: 1706.03762 [cs.CL].

[Wu+22] Wu, H., Wen, C., Li, W., Li, X., Yang, R., and Wang, C. *Transformation-Equivariant 3D Object Detection for Autonomous Driving*. 2022. arXiv: 2211.11962 [cs.CV].

[YML18] Yan, Y., Mao, Y., and Li, B. "SECOND: Sparsely Embedded Convolutional Detection". In: *Sensors* 18.10 (2018). ISSN: 1424-8220. DOI: 10.3390/s18103337. URL: https://www.mdpi.com/1424-8220/18/10/3337.

[Yu+22] Yu, H., Luo, Y., Shu, M., Huo, Y., Yang, Z., Shi, Y., Guo, Z., Li, H., Hu, X., Yuan, J., and Nie, Z. *DAIR-V2X: A Large-Scale Dataset for Vehicle-Infrastructure Cooperative 3D Object Detection*. 2022. arXiv: 2204.05575 [cs.CV].

[ZT17] Zhou, Y. and Tuzel, O. *VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection*. 2017. arXiv: 1711.06396 [cs.CV].

[Zim+23] Zimmer, W., Creß, C., Nguyen, H. T., and Knoll, A. C. *A9 Intersection Dataset: All You Need for Urban 3D Camera-LiDAR Roadside Perception*. 2023. arXiv: 2306.09266 [cs.CV].