Hybrid Data for Robust Object Detection in Autonomous Driving

Elisa Hagensieker

University of Osnabrueck, 49074 Osnabrueck, Germany
ehagensieker@uni-osnabrueck.de

Abstract. Object detection is essential in autonomous driving as it allows vehicles to accurately perceive and understand their surroundings. This study investigates the effectiveness of using both real-world and synthetic datasets, known as hybrid data, to improve object detection models. Our approach involves incorporating virtual KITTI 2 into training a pre-trained model and comparing its performance to a model trained only on real-world data. Through evaluating our model on both the KITTI and Virtual KITTI 2 datasets, we aim to provide recommendations for future training. Our findings indicate that ...

Keywords: Autonomous Driving · Deep Learning · Hybrid Data · KITTI · Object Detection · Virtual KITTI.

1 Introduction

1.1 Background and Motivation

The rapid advancement of autonomous driving technology has led to an increased focus on developing reliable and accurate object detection systems [1]. These systems are essential for the safe navigation of autonomous vehicles. They allow the vehicle to perceive and interpret its surroundings by identifying real-time obstacles such as pedestrians, other vehicles, and road infrastructure. Among the different techniques used for object detection, monocular object detection stands out due to its cost-effectiveness and simplicity. This technique relies on a single camera rather than complex multi-sensor setups [17].

One potential way to improve the performance of monocular object detection is by integrating synthetic data into the training process. Synthetic data, created using computer simulations or GANs, offers certain advantages. It enables the creation of large, diverse, and highly controlled datasets that can be customized for specific training requirements. This approach reduces the costs and time associated with annotating the datasets.

1.2 Problem Statement

This project aims to test the benefits of using synthetic data, the Virtual KITTI 2 dataset [4] to improve the overall performance and robustness of the detection

model. By exposing the model to a wide range of simulated environments and conditions, including varying lighting and weather, we expect the model to be better prepared to handle unseen real-world situations. Additionally, synthetic data will accelerate the development and testing cycles of autonomous driving systems, as it is not limited by the complex process of collecting and annotating real-world data.

1.3 Objectives

First, we will discuss the significance of object detection in autonomous driving, introducing the concept using synthetic and real-world data. Then, we will present the datasets, preprocessing steps, model architecture, and training process, including evaluation metrics. The experiments section will outline the different data configurations tested and the criteria for comparing the performance of these models. The performance of the different models will be analyzed and compared in the Results and Analysis section, followed by a discussion of the project's limitations. Lastly, we will summarize the key learnings and reflections from the project, highlighting the impact of hybrid datasets and outlining areas for future work.

2 Related Work

2.1 Autonomous Driving

Autonomous driving technologies are transforming the automotive industry, improving safety, efficiency, and convenience. These systems minimize human error and optimize vehicle operation, ultimately creating safer roads and more efficient driving experiences.

The Society of Automotive Engineers (SAE) defines levels of driving automation from Level 0 (no automation) to Level 5 (full automation) [6]. Object detection plays a crucial role across these levels, enabling the vehicle to perceive and interpret its surroundings, identify obstacles, and make real-time driving decisions. Advanced object detection systems are essential for achieving higher levels of autonomy, particularly in complex environments.

In 2022, Mercedes-Benz became the first automaker to receive international approval to manufacture a car capable of Level 3 autonomous driving. However, the usage of this technology is limited. For example, the Mercedes S-Class with Drive Pilot can only be used in specific conditions, such as daytime highway driving at speeds below 60 kilometers per hour [1]. Training the models on diverse datasets could help extend the environmental conditions in which the technology can be utilized.

2.2 Object Detection

In autonomous driving and object detection, advancements have been made by diverse methodologies aiming to enhance model accuracy and robustness. One

approach is the CIE model [19], which integrates contextual information with explicit (geometric properties) and implicit features (learned image features) to improve 3D object detection performance. Two baseline models are used: V2-99 [9] for 2D image feature extraction and depth key handling, and PLUMENet [18] for 3D space handling of bounding boxes and classification. Key components include an orientation-aware image backbone to align features and enhance object orientation understanding. Local Ray Attention transforms 2D features into 3D voxel representations in the camera's coordinate system, crucial for accurate spatial mapping. The 3D BEV Backbone predicts bounding boxes and generates a 3D occupancy map using PLUMENet. Augmentation techniques like random world rotation and flipping are utilized. The model is evaluated through Average-Precision, showcasing its top-ranking performance in 2022.

2.3 Hybrid Data

Benefiting from low acquisition costs, learning from synthesized data is an attractive way to scale up training data. However, these methods often depend on pre-built 3D assets to simulate real-world environments, which involves significant effort. Generative models can also be considered a neural rendering alternative to graphics engines. GANs train a generator network to produce data indistinguishable from real examples, as judged by a discriminator. Examples include BigDatasetGAN, which generates classification datasets by conditioning the generation process on category labels, and SSOD, which designs a GAN-based generator for synthesizing images with 2D bounding box annotations for object detection tasks.

The project in [10] introduces a novel approach to generate high-resolution 3D training data using GANs. Lift3D extends the capabilities of StyleGAN2 into the domain of 3D by integrating a neural radiation field (NeRF). This enables the synthesis of photorealistic and 3D-consistent images. Key features of their approach include disentangling latent spaces to control pose and appearance separately and optimizing NeRF to lift 2D images into 3D. It significantly boosts object detection model performance through realistic 3D annotations, outperforming traditional methods like ShapeNet, and improving deployment readiness across varied environments. Lift3D addresses dataset scarcity challenges, accelerating autonomous driving research cycles.

When using GANs for generating synthetic data in applications such as autonomous driving, a significant challenge is the "Reality Gap." GANs struggle to replicate the complexity, variability, and class distribution found in real-world datasets. The use of GAN-generated data raises ethical and safety concerns, especially in safety-critical domains like autonomous driving, due to uncertainties about how well GAN-generated data represents real-world complexities. GANs may produce visually plausible but semantically incoherent scenes, leading to unrealistic or misleading training scenarios.

3 Methodology

3.1 Data

A wide variety of datasets are available in the field of autonomous driving, such as Waymo [14], Lyft L5 [11], nuScenes [3], SYNTHIA [12], KITTI [5], and RELLIS 3D [7]. To simplify the process of matching the shape and annotation formats of the datasets, we decided to use KITTI and virtual KITTI due to their compatibility.

Fig. 1. Comparison of the datasets: Top row: KITTI images. Middle row: Virtual KITTI 1.3.1 images. Bottom row: Virtual KITTI 2.0 images. [2]

KITTI The KITTI dataset is a comprehensive set of benchmarks created to advance the field of autonomous driving. It contains data from one LiDAR sensor, four cameras, and one localization system, covering 22 different scenes over about 1.5 hours of driving. The dataset includes 80,000 2D and 3D bounding boxes, captured over a driving distance of 73.7 km. It encompasses eight different classes, with the "car" class appearing in about 66 percent of various driving instances. The sensors are accurately calibrated and synchronized to provide precise ground truth data for tasks like stereo vision, optical flow, visual odometry/SLAM, and object detection. The dataset's realism, complexity, and numerous classes present significant challenges and important benchmarks for autonomous driving research.

Virtual KITTI 2 The Virtual KITTI 2 dataset [2] leverages cutting-edge techniques in computer graphics to create a comprehensive synthetic video dataset. Developed using the Unity game engine 2018.4 LTS and High Definition Render

Pipeline (HDRP), it offers a diverse range of photorealistic scenes that simulate real-world conditions such as varying lighting and weather scenarios. Figure 2 shows six conditions: sunset, fog, RGB, original, rain, and morning. It addresses the substantial challenges of acquiring and annotating large-scale video datasets by providing fully labeled data for tasks including object detection, tracking, scene segmentation, depth estimation, and optical flow.

We chose Virtual KITTI 2 over Virtual KITTI 1 [4] because it provides more realistic lighting, improved shadows, richer annotations, and utilizes recent enhancements in lighting and post-processing, bridging the gap between Virtual KITTI and KITTI images (see Figure 1 for examples).

This dataset contains 50 scenes based on 5 original KITTI scenes, each featuring 10 variations to capture different environmental conditions. It consists of approximately 3 hours of synthetic video data and includes over 540,000 annotations of 2D and 3D bounding boxes for cars. Virtual KITTI 2 allows for experiments that compare real and synthetic data, enabling insights into algorithm performance under controlled conditions and varied imaging challenges [2].

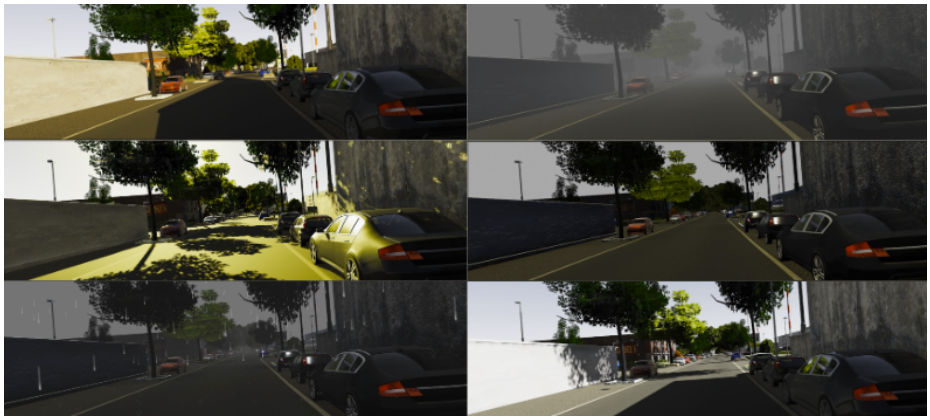

Fig. 2. Samples from the same scene with different variations.
Upper left: sunset **Upper right:** fog
Middle left: RGB **Middle right:** original
Lower left: rain **Lower right:** morning

3.2 Data preprocessing

Several steps were taken to prepare the KITTI and Virtual KITTI 2 datasets for training. These steps are contained in preprocessing.py and are detailed next:

1. Resizing: The images in the dataset were resized to match a specified input shape, ensuring consistency across all input data.

2. Normalization: The pixel values of the images were normalized to a range of 0 to 1, which helps improve the performance and stability of the neural network during training.
3. Padding: In cases where images or annotations varied in size, padding was applied to standardize the dimensions, allowing for uniform processing within the neural network. Bounding boxes were added as background boxes with another label to differentiate between background and car instances.
4. Shuffling: The dataset was shuffled before training to ensure that the data is fed to the model in a random order. This prevents the model from learning patterns specific to the order of the data, which could lead to overfitting and bad generalization abilities.
5. Batching: The data was organized into batches of a fixed size. This allows the model to process multiple images simultaneously, making the training process more efficient.

The data loading function ensured that these preprocessing steps were consistently applied to both training and validation datasets. Different formats for the bounding boxes and the folder structure require separate functions for KITTI and Virtual KITTI 2.

3.3 Model Architecture

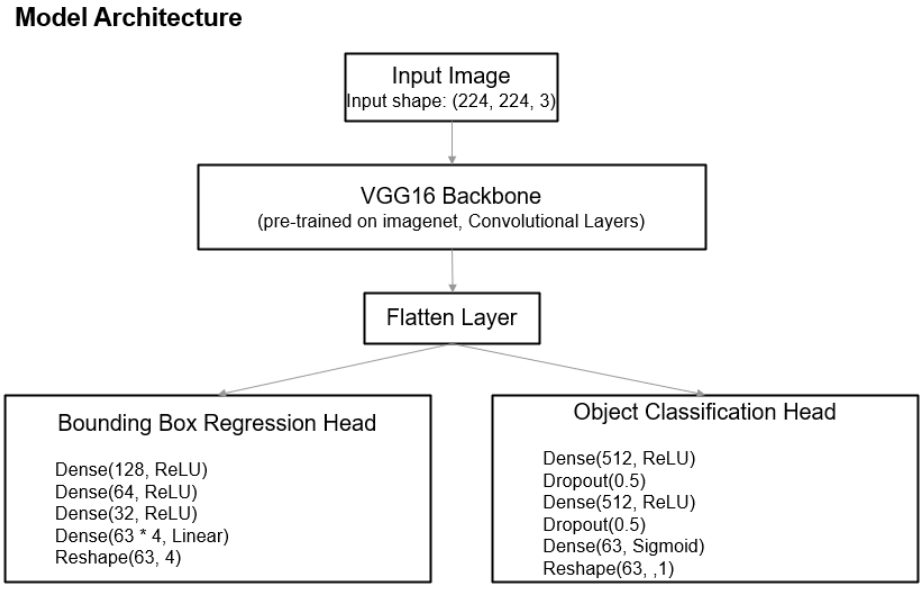

Fig. 3. Architecture of the Model

The model, as shown in 3, is based on the pre-trained VGG16 network, which is typically utilized for image classification tasks [13]. VGG16 provides a robust foundation for feature extraction, owing to its deep convolutional layers that capture high-level visual features. To adapt the model to predict both bounding boxes and labels, additional layers were added to perform bounding box regression and classification tasks. First, the output from VGG16 is flattened and then fed into the newly added output heads. The model architecture includes two distinct heads:

1. Bounding Box Regression Head (3 left path): The output head has four nodes, each corresponding to an edge of the image. Values are normalized to fall within the range [0, 1] relative to the image dimensions. The activation function used for these output nodes is sigmoid [8]. To optimize the network for this regression task, Mean Squared Error (MSE) is used as the loss function. MSE effectively penalizes the squared differences between the predicted and actual bounding box coordinates [15].
2. Object Classification Head (3 right path): The second output head typically consists of a single output node when performing binary classification. For this node, the sigmoid activation function is used again. The Binary Crossentropy loss function is employed here. It measures the difference between the predicted probability and the actual class label [15].

By fine-tuning the model, we aim to specialize it for a specific use case. Due to time constraints, only 2D object detection was performed, which limited the scope of the project to detecting and classifying objects within two-dimensional image space.

3.4 Evaluation Matrices

We used a combination of evaluation metrics to assess our model's performance, ensuring accurate label classification and bounding box accuracy [16].

Precision and **Recall** are widely accepted metrics used to evaluate the performance of label classifications. Precision indicates the accuracy of positive predictions made by the model and is calculated as the ratio of correctly predicted positive observations to the total predicted positives. In our project, high precision ensures that objects classified as vehicles are indeed cars, minimizing false alarms. Meanwhile, high recall in the context of autonomous driving ensures that the model detects as many cars as possible, reducing the likelihood of accidents.

To assess the precision of the predicted bounding boxes, the model used two metrics: **Binary Intersection Over Union** (IoU) and **Mean Average Error** (MAE). MAE calculates the average magnitude of errors in the predicted bounding box coordinates. It is defined as:

$$\text{MAE} = \frac{1}{n} \sum_{n=0}^{n} |\text{Predicted Value\_}i - \text{Actual Value\_}i|$$

Here, n represents the total number of predictions, and the sum indicates the absolute differences between the predicted and actual bounding box coordinates. A lower MAE indicates a more accurate bounding box prediction.

IoU is a commonly used metric to evaluate the accuracy of predicted bounding boxes in object detection tasks. It calculates the intersection over union between the predicted bounding box and the ground truth bounding box. IoU is defined as:

$$\text{IoU} = \frac{\text{Area of overlap}}{\text{Area of Union}}$$

The Area of Overlap is where predicted and ground truth bounding boxes intersect, while the Area of Union covers the total area of both bounding boxes. The IoU value ranges from 0 to 1, with 1 indicating a perfect overlap. Typically, a threshold of 0.5 or 0.7 is used; if the IoU exceeds this threshold, the prediction is considered accurate. We only focus on the bounding boxes associated with the target class ID 0 (cars) to measure the accuracy of car detections, which is crucial for assessing the model's effectiveness in autonomous driving scenarios. The background bounding boxes are not important for our use case.

Mean Average Precision (MAP) is a widely used metric for evaluating the overall performance of object detection models. It is calculated by averaging the Average Precision (AP) scores for each class in the dataset. The AP is obtained from the precision-recall curve, and the MAP is the average of these AP values across all classes. However, due to the lack of a TensorFlow implementation and time constraints, it was not utilized in this project.

4 Experiments

4.1 Experimental Setup

The training was conducted on Google Colab Pro using a T4 GPU. Additionally, the code and datasets are available on GitHub `https://github.com/elihagen/EnhancingAI_Project.git` and on the high-performance computing cluster of the University of Osnabrück. The model was trained using a combination of synthetic and real-world datasets. The training process included several key aspects. Initially, the training parameters were defined, with the use of the Adam optimizer and a learning rate of 1e-4. The training ran for 25 epochs with a batch size of 32. Regular checkpointing was implemented to track progress and ensure the training process was secure. Model checkpoints were saved to a CSV file, allowing for monitoring of the model's performance and preventing data loss. This also ensured that training could be resumed seamlessly if interrupted.

Results - KITTI dataset When trained solely on the KITTI dataset, the model demonstrated strong performance in recognizing and localizing objects in real-world scenarios. The results indicate a high mAP.

Results - Combined Data Set

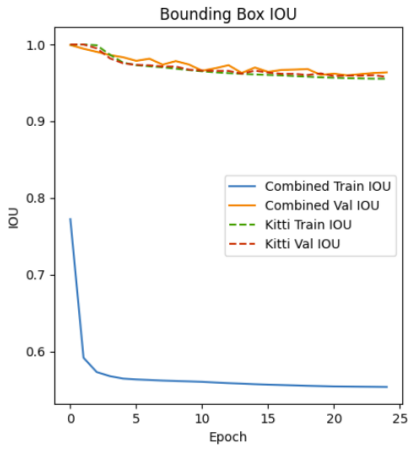

Fig. 4. Bounding Box IOU

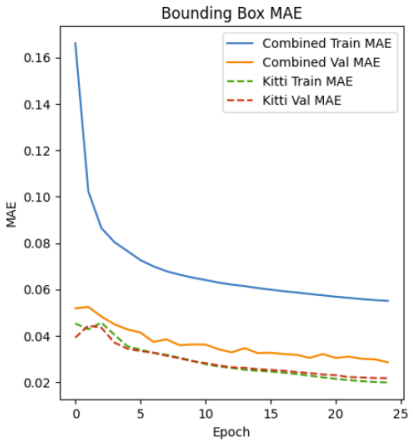

Fig. 5. Bounding Box Mean Average Error

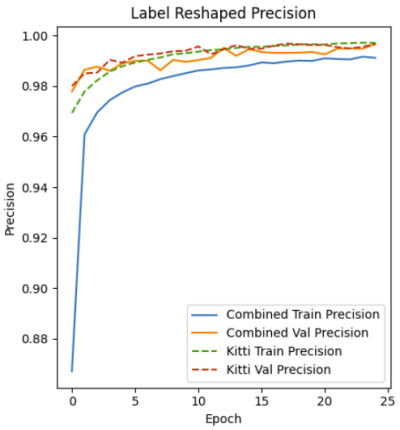

Fig. 6. Label Precision

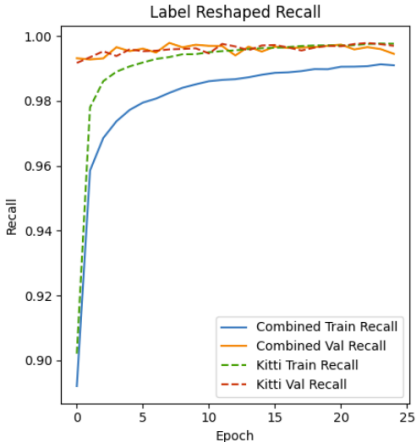

Fig. 7. Label Recall

5 Discussion

5.1 Interpretation of Results

Subsection discussing how plausible the results are. Subsection putting and discussing our results in the perspective of existing literature - did we improve compared to other models with real data only?

Despite the promising results, our project has several limitations:

Dataset Imbalance: The KITTI dataset includes multiple object classes, while the Virtual KITTI dataset focuses solely on cars. This imbalance may affect the model's performance in detecting other object classes. Reality Gap: The

synthetic data, although photorealistic, may not perfectly capture the complexity and variability of real-world scenarios. Resource Constraints: Due to limited computational resources and time constraints, our experiments may not have fully explored the potential of synthetic data. Model constraints: It is important to note that this approach might not utilize the best-pretrained model and was chosen due to the time constraints of the course.

6 Conclusion

6.1 Learnings and Reflections

Initially, we encountered an issue with inverted y-coordinates in the bounding boxes of the images. This discrepancy highlighted the importance of correctly interpreting and using the datasets provided for KITTI and Virtual KITTI and ensuring that the bounding boxes aligned accurately with the images was crucial for the model's ability to detect objects properly.

Another significant challenge was dealing with the multi-class problem in the KITTI dataset, which includes various object classes, while the Virtual KITTI dataset focused solely on cars. To maintain consistency and simplicity in our experiments, we ignored the other classes in the KITTI dataset and focused exclusively on car detection. This decision allowed us to streamline our model and dataset, reducing complexity.

When working on object detection with multiple objects in images, we had to address the challenge of padding and defining the number of output nodes in the model. We resolved this by implementing padding and assuming a maximum of 16 objects per image based on the KITTI dataset. This approach also handled images without cars by treating them as a binary problem with a single label for non-car bounding boxes.

Additionally, we explored the complexity of 2D versus 3D object detection. Adapting the camera coordinate system to the vehicle coordinate system for 3D object detection proved to be a complex task. Due to the complexities associated with 3D detection, we decided to concentrate on 2D object detection while keeping open the possibility of testing 3D detection as well. The model and preprocessing code for 3D detection can be found in our GitHub repository, enabling future exploration and experimentation in this area.

6.2 Future Work

Future work could focus on addressing the limitations identified in this project. Potential directions include:

Expanding Object Classes: Incorporating a wider range of object classes in the synthetic dataset to match the diversity of real-world scenarios. Reducing the Reality Gap: Improving the realism of synthetic data using advanced techniques such as neural rendering and GANs to better replicate real-world complexities. Enhancing Model Architecture: Exploring more sophisticated model architectures and training strategies to further boost detection performance. Extensive

Hyperparameter Tuning: Conducting comprehensive hyperparameter optimization to maximize the model's performance. Adaptive Dataset Expansion: One could also consider to gradually integrate subsets of synthetic data into the KITTI dataset during each training epoch to help the model adapt to variability and improve generalization and real-world detection performance.

References

1. Aalst, W.v.d.: Six levels of autonomous process execution management (apem)
2. Cabon, Y., Murray, N., Humenberger, M.: Virtual kitti 2. arXiv preprint arXiv:2001.10773 (2020)
3. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11621–11631 (2020)
4. Gaidon, A., Wang, Q., Cabon, Y., Vig, E.: Virtual worlds as proxy for multi-object tracking analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4340–4349 (2016)
5. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: 2012 IEEE conference on computer vision and pattern recognition. pp. 3354–3361. IEEE (2012)
6. International, S.: Sae levels of driving automation™ refined for clarity and international audience, `https://www.sae.org/blog/sae-j3016-update`
7. Jiang, P., Osteen, P., Wigness, M., Saripalli, S.: Rellis-3d dataset: Data, benchmarks and analysis. In: 2021 IEEE international conference on robotics and automation (ICRA). pp. 1110–1116. IEEE (2021)
8. Keras: Layer activation functions, `https://keras.io/api/layers/activations/`
9. Lee, Y., Park, J.: Centermask: Real-time anchor-free instance segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 13906–13915 (2020)
10. Li, L., Lian, Q., Wang, L., Ma, N., Chen, Y.C.: Lift3d: Synthesize 3d training data by lifting 2d gan to 3d generative radiance field. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 332–341 (2023)
11. Mandal, S., Biswas, S., Balas, V.E., Shaw, R.N., Ghosh, A.: Lyft 3d object detection for autonomous vehicles. In: Artificial Intelligence for Future Generation Robotics, pp. 119–136. Elsevier (2021)
12. Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.M.: The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3234–3243 (2016)
13. Sanchez, S., Romero, H., Morales, A.: A review: Comparison of performance metrics of pretrained models for object detection using the tensorflow framework. In: IOP conference series: materials science and engineering. vol. 844, p. 012024. IOP Publishing (2020)
14. Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al.: Scalability in perception for autonomous driving: Waymo open dataset. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2446–2454 (2020)

15. TensorFlow: Module: tf.keras.losses, https://www.tensorflow.org/api_docs/python/tf/keras/losses
16. TensorFlow: Module: tf.keras.metrics, https://www.tensorflow.org/api_docs/python/tf/keras/metrics
17. Vargas, J., Alsweiss, S., Toker, O., Razdan, R., Santos, J.: An overview of autonomous vehicles sensors and their vulnerability to weather conditions. Sensors **21**(16), 5397 (2021)
18. Wang, Y., Yang, B., Hu, R., Liang, M., Urtasun, R.: Plumenet: Efficient 3d object detection from stereo images. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 3383–3390. IEEE (2021)
19. Ye, Q., Jiang, L., Zhen, W., Du, Y.: Consistency of implicit and explicit features matters for monocular 3d object detection. arXiv preprint arXiv:2207.07933 (2022)