ISSN 2616-5775 Vol. 7, Issue 8: 63-73, DOI: 10.25236/AJCIS.2024.070810

Optimization of Multi-View 3D Object Detection in Urban Traffic Environments Based on the PETR Algorithm

Bohan Zhang[1,*]

[1]*The High School Affiliated to Renmin University of China, Beijing, China*
[*]*Corresponding author*

Abstract: *In the intricate environment of urban centers, autonomous vehicles encounter multifaceted perceptual challenges. Traditional 2D object detection methods fail to accurately provide the feature information of 3D objects, leading to the failure of object identification and behavior prediction. The PETR algorithm improves this problem through multi-vision 3D object detection. This thesis is dedicated to optimizing the PETR algorithm from three perspectives to elevate the performance and efficacy of 3D object detection. These optimizations include refining the network backbone, adjusting image input parameters, and enhancing the training parameters such as the learning rate, gradient clipping parameters, and the choice of optimizer. Utilizing the NuScenes Dataset for model training, the final evaluation and comparison of model performance are mainly based on mAP metric.*

Keywords: *Object Detection, Urban Traffic, PETR Algorithm*

1. Introduction

As autonomous driving technology advances rapidly, it is particularly important to accurately perceive the surrounding environment of vehicles and detect objects. Autonomous driving has numerous hidden issues and risks to be addressed. In the domain of 3D imaging, multi-view 3D object detection stands as a pivotal part.

Traditional 2D object detection methods fail to accurately provide the feature information of 3D objects, leading to the failure of object identification and behavior prediction. Multi-view 3D object detection, however, offers an effective solution, enabling vehicles to perceive the surrounding environment more accurately. Multi-view object detection captures environmental information from various aspects, thereby generating a more comprehensive and precise 3D object model. This means that the detection system can still accurately detect and recognize objects even when multiple objects cover each other. Furthermore, through the fusion of multiple views, a 3D coordinate system can be established, integrating 2D features from different angles with 3D information. As a result, it can track and predict target behaviors more accurately during real-time object detection.

Nevertheless, the vast amount of data involved in multi-view 3D object detection demands sophisticated algorithms and models capable of efficiently handling and analyzing extensive image datasets. An applicable feature extraction model (backbone) is essential for the achievement of behavior prediction and object classification. To overcome these problems, this thesis will explore deep learning-based network architectures and optimization techniques to enhance the performance and efficacy of 3D object detection.

Leveraging a multi-view approach and PETR algorithm, experiments and optimizations on the NuScenes dataset for 3D object detection based on PETR (Position Embedding Transformation) are carried out. By extracting features from multi-view 2D images with a backbone, and fusing them with the 3D spatial features of the object, it allows vehicles to acquire the capability for 3D perception. The thesis aims to identify the optimal backbone for the PETR algorithm and optimize its model parameters to improve the accuracy and performance of object detection.

2. Problem Analysis

In the intricate environment of urban centers, autonomous vehicles encounter multifaceted perceptual

ISSN 2616-5775 Vol. 7, Issue 8: 63-73, DOI: 10.25236/AJCIS.2024.070810

challenges. It is necessary to focus on multiple dynamic and static objects at the same time, including other vehicles, pedestrians, traffic signals, and unexpected obstacles (such as bicycles or scooters). In such a complex situation, traditional 2D vision system are prone to visual blindness, which may result in misjudgments and omissions. For instance, a pedestrian crossing the road from the side of the vehicle may not be captured or analyzed accurately by a 2D camera. Such visual blindness can cause severe traffic safety issues.

The forthcoming sections will analyze the occlusion challenges inherent in object detection and the limitations of traditional 2D detection systems in accurately localizing and predicting object behavior.

2.1. Occlusion Issues

Occlusion issues can be divided into two categories:

1) Inter-class occlusion, where the target is obscured by objects of a different class. For example, a pedestrian walking a dog may have their lower body partially hidden by the dog.

2) Intra-class occlusion, where the target object is obscured by objects of the same class. In our case, it means the pedestrian is hidden by another person.

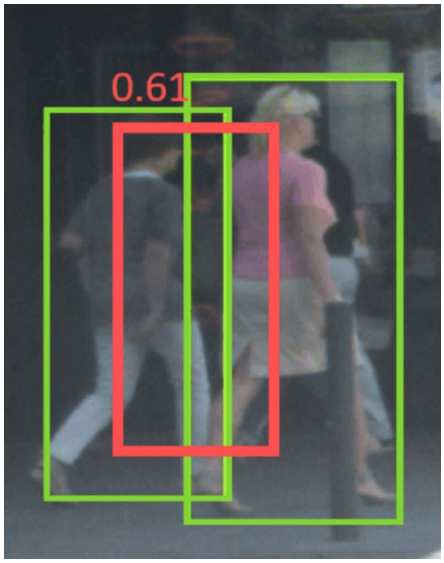

Figure 1: Occlusion Issues

Both intra-class and inter-class occlusions pose challenges for traditional 2D vision systems. The first issue arises in pedestrian detection, where individuals standing close to each other are often misjudged as one person (in Figure 1, the green boxes represent the ideal correct prediction boxes, while the red box refers to the actual prediction box). When the prediction boxes of two similar objects are in close proximity, one may be suppressed by NMS, leading to omissions in pedestrian detection. Adjusting the NMS threshold has significant limitations when solving such issues. A low threshold may lead to omissions, while a high threshold might cause false positives as well. Additionally, different scenarios may require different thresholds. Most people suggest to solve this problem by optimizing the loss function. However, a multi-view approach can offer a more straightforward solution. Even if occlusion occurs in one direction, multi-view approach can prevent omissions through information from other angles[1].

The second issue pertains to the inability of 2D vision to effectively obtain depth information and accurately determine the distance. In contrast, 3D vision and multi-view cones can generate a 3D space to extract depth information and accurately generate a 3D bounding box.

Taking Centernet (a 2D vision algorithm) as an example, the deficiencies in various occlusion scenarios (ABCD) are discussed as follows:

A: The object is at the edge of the image and partially visible

2D vision often fails to detect the object when it is at the edge of the image and partially visible. As shown in Figure 2, the cars circled in red are undetected.

ISSN 2616-5775 Vol. 7, Issue 8: 63-73, DOI: 10.25236/AJCIS.2024.070810

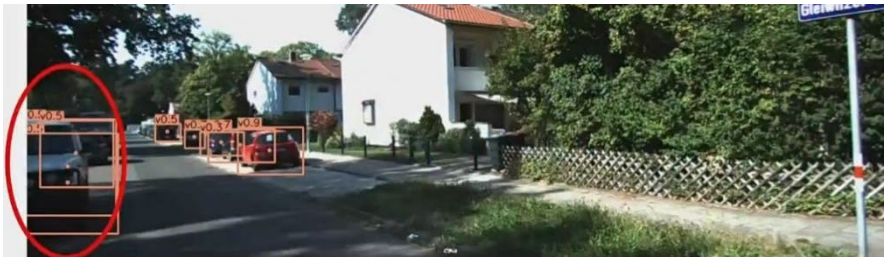

Figure 2: Partially visible cars undetected by 2D vision

How multi-view 3D vision systems address this issue: Multi-view detection is adopted to identify the complete object.

B: Omissions: The object is far away with small size in the image

As shown in Figure 3, the distant cars circled in red are undetected.

Figure 3: Distant cars undetected by 2D vision

C: Omissions: Multiple targets are clustered together.

As shown in Figure 4, the individuals circled in red are undetected due to object overlap.

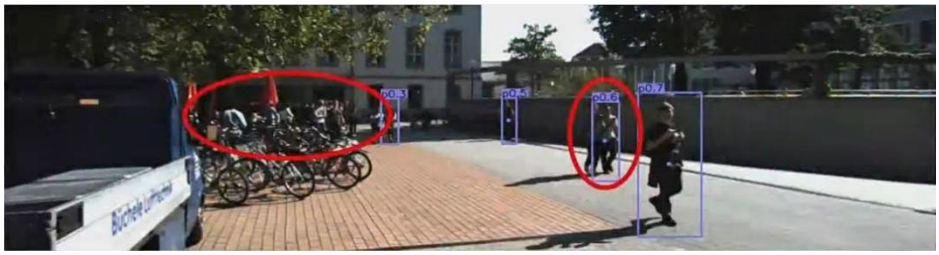

Figure 4: Individuals undetected by 2D vision due to object overlap

How multi-view 3D vision systems address this issue:

1) Even if occlusion occurs in one direction, multi-view approach can prevent omissions through information from other angles.

2) Since 3D vision can perceive distance, it allows for the distinction between objects that appear to overlap in 2D images. Through encoded 3D information, it can tell the different distances from the car (in the coordinate system centered on the car generated by the multi-view cones), thus identifying them as separate entities.

D: Omissions and false detection: Rare objects in the Dataset (classification issues)

In Figure 5, there is an omission in identify people. In Figure 6, a picture on the wall is misidentified as a person.

ISSN 2616-5775 Vol. 7, Issue 8: 63-73, DOI: 10.25236/AJCIS.2024.070810

Figure 5: People failed to be identified by 2D vision

Figure 6: Picture on the wall misidentified as a person by 2D vision

Regarding Figure 5, since most training examples feature standing people, the 2D vision system cannot correctly "classify" the situation. Meanwhile, 2D vision system cannot utilize depth information for prediction neither.

Regarding Figure 6, since the 2D vision system cannot use depth information for prediction, it leads to an inability to distinguish between 3D objects and 2D objects and the wrong identification of plane objects as 3D entities. However, through depth information, 3D object detection is available to discern that there is a picture on the wall rather than a 3D object.

2.2. Failure in Behavior Positioning or Prediction

Here's a comparison of the differences in positioning between 2D vision and 3D vision.

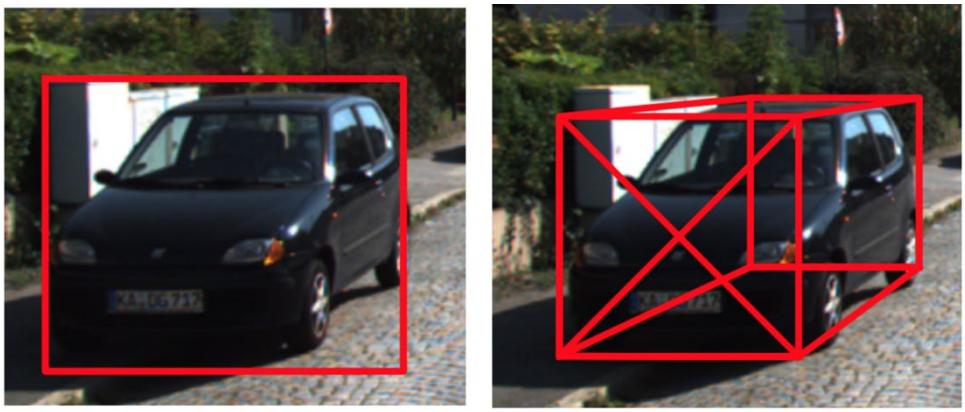

Figure 7: 2D vision positioning (left) and 3D vision positioning (right)

First of all, 2D object detection can only achieve positioning on the target plane (as shown on the left side of Figure 7). Traditional 2D object detection generates 2D bounding boxes without providing depth

ISSN 2616-5775 Vol. 7, Issue 8: 63-73, DOI: 10.25236/AJCIS.2024.070810

information. Through multi-view cones, 3D space can be generated, extracting depth information and creating 3D bounding boxes. Depth information, as part of the feature set, is of great help to in more accurate "classification". For instance, 2D vision might identify a picture of a car as a realistic car. However, 3D vision won't make such a mistake since the depth information (size) is incompatible with an actual vehicle.

The following is a comparison of the differences in behavior prediction between 2D vision and 3D vision.

Figure 8: Vehicle behavior in 2D vision (left) and vehicle behavior in 3D vision (right)

According to *[5]*, it is difficult to estimate the overall pose of an object only based on the detected 2D bounding box. As shown in Figure 8, when a car is moving straight ahead, its overall pose is moving forward in 3D vision, while its pose or vehicle angle is constantly changing in 2D bounding boxes.

3. Materials and Methods

3.1. Dataset

This project utilized the NuScenes dataset, a public large-scale dataset for autonomous driving developed by the Motional team. The dataset records 1,000 driving scenarios from Boston and Singapore, where are known for heavy traffic. Multiple devices such as 6 cameras, 1 LIDAR, 5 RADARs, a GPS, and an IMU are included in this dataset. It covers about 1.4 million camera images, 390,000 LIDAR scans, 1.4 million RADAR scans, and bounding boxes of approximately 1.4 million objects in 40,000 key frames. In the real labels of the dataset, 23 object categories are marked with 3D bounding boxes at a frequency of 2Hz. As is shown in Figure 9.

Figure 9: Rear camera picture in NuScenes dataset (taken from dataset)

ISSN 2616-5775 Vol. 7, Issue 8: 63-73, DOI: 10.25236/AJCIS.2024.070810

3.2. Introduction to the algorithm

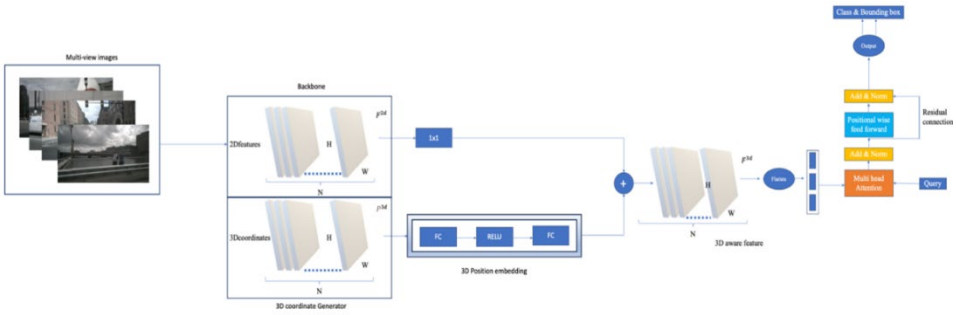

Figure 10: PETR Algorithm Flowchart

Initially, multi-view images are input into the backbone network (Residual Network) to extract multi-view 2D image features. At the same time, in the 3D Coordinate Generator, a 3D coordinate system is constructed by using the camera frustum space shared by all viewing angles. Based on the internal parameters of the camera, coordinates in the 3D world space can be obtained. After that, the 3D position encoder will generate 3D position-aware features with the combination of 2D image features and the 3D coordinates. Object queries generated by the query generator interact with and update the 3D position-aware features within the Transformer Decoder. The updated "query" is further used to predict 3D bounding boxes and object categories. As is shown in Figure 10.

3.2.1. 3D Coordinate Generator

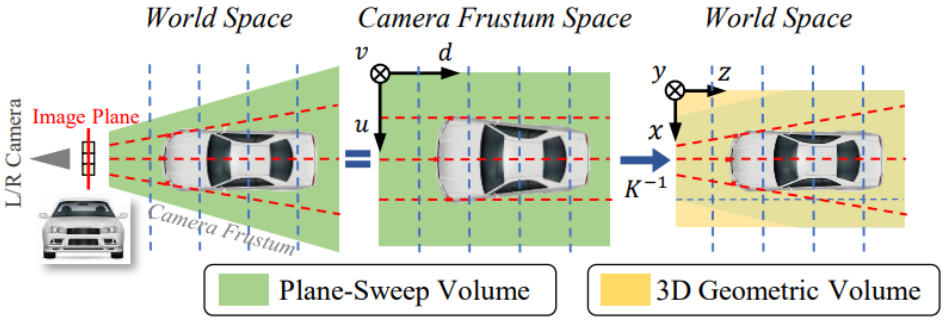

Figure 11: 3D Coordinate Generator [2]

The (DGSN) 3D Coordinate Generator function is to establish a connection between 2D images and 3D coordinates. Through the 3D Generator, the points in the camera frustum space can be projected into 3D space. As is shown in Figure 11.

The process of the 3D Coordinate Generator is as follows:

The first step is to define the camera frustum space. The camera frustum space is denoted by (u, v, d), and the 3D world space is denoted by (x, y, z). Camera internal parameters can transform the camera frustum space into the 3D world coordinate system. In the "mesh grid," each point can be represented as:

$$p_{i,j}^{3d} = (x_{i,j}, y_{i,j}, z_{i,j}, 1)^T \tag{1}$$

(uj, vj) refers to the pixel coordinates in the image.

Since there are 6 camera views in the NuScenes dataset, there are overlapping regions between different cameras. Considering that a 3D point may be located in the visual space of multiple cameras, in the case of multiple cameras, the coordinates can be expressed as:

Calculation method: Multiplying the inverse of the transformation matrix by the coordinates of each point in the "mesh grid."

Calculation formula:

$$p_{i,j}^{3d} = K_i^{-1} p_j^m \tag{2}$$

Finally, the "Global" coordinate points are normalized to specify the spatial range.

ISSN 2616-5775 Vol. 7, Issue 8: 63-73, DOI: 10.25236/AJCIS.2024.070810

Normalization formula:

$$\begin{cases} x_{i,j} = (x_{i,j} - x_{min})/(x_{max} - x_{min}) \\ y_{i,j} = (y_{i,j} - y_{min})/(y_{max} - y_{min}) \\ z_{i,j} = (z_{i,j} - z_{min})/(z_{max} - z_{min}) \end{cases} \tag{3}$$

Reasons for normalization:

1) Normalization can ensure that the input data is within the same scale range (such as [0, 1]), avoid excessive differences in pixel values between different images. Meanwhile, it is conducive to improving the convergence speed of the model and making learning image features more effective.

2) Normalization can prevent gradient explosion and gradient disappearance. If the value is excessively large or small, it may lead to severe gradient changes. With the help of normalization, it can improve gradient stability.

3) Normalization can accelerate the training process. By specifying the range of data, it can reduce the computational complexity to speed up model training[2-4].

3.2.2. Query

Object query is a vector used to query the features of the target. Through querying, it can mark and track the target. In initialization phase, Query will be given pre-trained parameters and pass through a simple MLP. Meanwhile, the query vector will search for locations on the feature map that are similar to the "features" based on existing parameters. Once the feature locations are obtained, the target position is determined, enabling the target to be marked and visualized. In M3DPET, the query can be iteratively updated by the decoder. The initial object query can be acquired by inputting 3D anchor coordinates (processed through an MLP and two linear layers).

3.2.3. Decoder

In the M3DPET algorithm, the Encoder-Decoder model framework is utilized. Its characterization is End-to-End learning algorithm (also known as Sequence to Sequence learning). Encoding is to transform the input "sequence" into a vector, while decoding is to convert the existing vector into a new sequence.

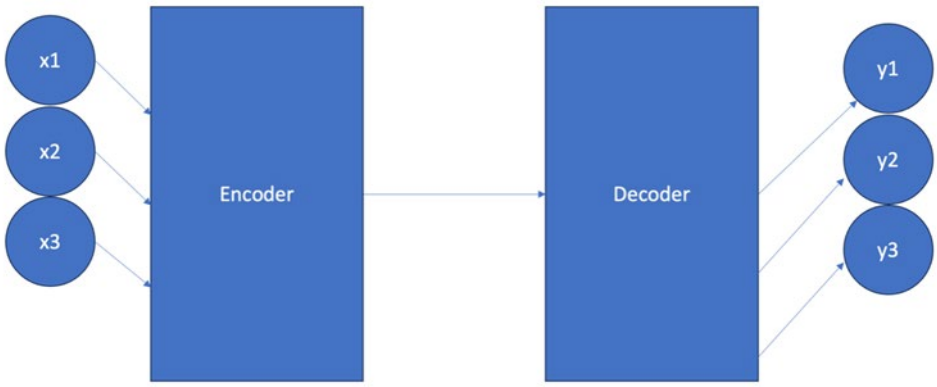

Figure 12: Encoder-Decoder Model Framework

The running process of the decoder layers can be divided into two parts. The first part is the combination of object queries and 3D positional features. The second part is the similarity comparison between updated object queries and the object at the target location found through the first part to determine the object type. As is shown in Figure 12.

The following formula represents the running process of the decoder layers:

$$Q_l = \Omega_l(F^{3d}, Q_{l-1}), \quad l = 1, \dots, L \tag{4}$$

The running process of the decoder can be divided into two parts. The first part is the combination of object query and 3D position features. The second part is the similarity comparison between updated object query and the object at the target location found through the first part to determine the object type. Ω_l represents the L-th layer of decoder; $\in$RM×C represents the updated object query; M and C are the number of channels and queries; F^{3d} represents 3D positional features.

ISSN 2616-5775 Vol. 7, Issue 8: 63-73, DOI: 10.25236/AJCIS.2024.070810

The formula demonstrates how each layer's decoder processes 3D positional features and object queries, with the object queries constantly updated during the process.

3.2.4. Encoder

As mentioned in the "decoder" section, Encoder converts the previously generated vector into an output sequence. In M3DPET, Encoder connects 2D feature vectors with 3D coordinates, and eventually obtains 3D features.

Here are the 2D feature vectors:

$$F_i^{3d} = \psi\left(F_i^{2d}, P_i^{3d}\right), \quad i = 1,2,\dots,N \tag{5}$$

$\psi\left(\mathrm{F_i^{2d}}, \mathrm{P_i^{3d}}\right)$ represents the processing method of encoder layers; $\mathrm{F_i^{2d}}$ refers to 2D features; $\mathrm{P_i^{3d}}$ refers to 3D location information.

3.3. Result Evaluation Metrics

True positives (TP): Positive samples correctly identified as positive.

True negatives (TN): Negative samples correctly identified as negative.

False positives (FP): False positive samples, namely, negative samples incorrectly identified as positive.

False negatives (FN): False negative samples, namely, positive samples incorrectly identified as negative.

3.3.1. Precision

$$Precision = \frac{TP}{TP+FP} \tag{6}$$

Precision represents the proportion of true positives among the images that have been identified.

3.3.2. Recall

$$Recall = \frac{TP}{TP+FN} \tag{7}$$

Recall represents the proportion of true positives in the test set that are correctly identified as positive.

3.3.3. AP (Average Precision) & mAP (mean Average Precision)

AP refers to the area under the precision-recall curve. Usually, the better a classifier, the higher the AP value. mAP refers to the average AP values across multiple categories. m refers to the arithmetic mean of the sum of various types of APs, namely, the value of mAP. The value of mAP must be in the range of [0,1]. Higher values indicate better model performance.

This thesis mainly uses the mAP metric to analyze the experimental results.

4. Data and Discussion

4.1. Experimental data

Based on the original Resnet implementation of the PETR algorithm, tests were conducted using the NuScenes dataset, and model performance tuning was performed on this basis. The performance of the PETR algorithm using Resnet is shown in Table 1:

Table 1: The performance of the PETR algorithm using Resnet

Backbone	mAP	mATE	mASE	mAOE	mAVE	mAAE:	NDS
Resnet	0.3137	0.8180	0.2755	0.6069	1.0077	0.2396	0.3599

1) Model performance tuning by modifying the Backbone:

By adjusting the original Backbone from Resnet to Vovnet for testing, the data is shown in Table 2.

Through comparison, it can be seen that the mAP value of Vovnet is 0.3677, which is higher than the original Resnet's performance of 0.3137, indicating a better performance.

2) Model performance tuning by modifying image input size:

ISSN 2616-5775 Vol. 7, Issue 8: 63-73, DOI: 10.25236/AJCIS.2024.070810

The previous test showed that the performance of Vovnet is better than that of Resnet. This step performs comparative testing by adjusting different image input sizes (1200 x 480, 800 x 320, 400 x 160) on the basis of Vovnet. The data is shown in Table 3.

Table 2: Vovnet result

Backbone	mAP	mATE	mASE	mAOE	mAVE	mAAE:	NDS
Resnet	0.3137	0.8180	0.2755	0.6069	1.0077	0.2396	0.3599
Vovnet	0.3677	0.7440	0.2701	0.4922	0.8234	0.2075	0.4301

Table 3: Different image input sizes

Backbone	Image input size	mAP	mATE	mASE	mAOE	mAVE	mAAE:	NDS
Vovnet	1200 x 480	0.2845	0.9037	0.2872	0.5570	0.9349	0.2331	0.3506
Vovnet	800 x 320	0.3677	0.7440	0.2701	0.4922	0.8234	0.2075	0.4301
Vovnet	400 x 160	0.0245	1.1905	0.4083	0.9457	1.2037	0.4393	0.1329

According to the data, the performance is best at an image input size of 800 x 320 under Vovnet.

3) Model performance tuning by modifying learning rate, optimizer, and other parameters:

With the highest mAP value based on Vovnet as the benchmark, further optimization testing of the model was conducted by adjusting the learning rate, optimizer, and gradient clipping parameters. The learning rate was adjusted from 2e-4 to 2e-5, the optimizer was changed to Momentum Gradient Descent, and the gradient clipping value (Max Norm) was adjusted from 35 to 30. The test data is shown in Table 4.

From the comparison data in Table 4, it is obviously that the optimized mAP value of Vovnet has increased from 0.3677 to 0.3794, further improving the performance.

Table 4: Results

Backbone	Parameters	mAP	mATE	mASE	mAOE	mAVE	mAAE	NDS
Vovnet	Original parameters: Learning rate: 2e-4 Optimizer: Adamw Gradient clipping:35	0.3677	0.7440	0.2701	0.4922	0.8234	0.2075	0.4301
Vovnet	Optimized parameters: Learning rate: 2e-5 Optimizer: Momentum Gradient Descent Gradient clipping:30	0.3794	0.7486	0.2714	0.4707	0.9266	0.2171	0.4262

4) Model performance tuning by modifying learning rate, optimizer, and other parameters:The optimized model showed a continuous decrease in training loss (Loss), with normal overall performance, as shown in Figure 13.

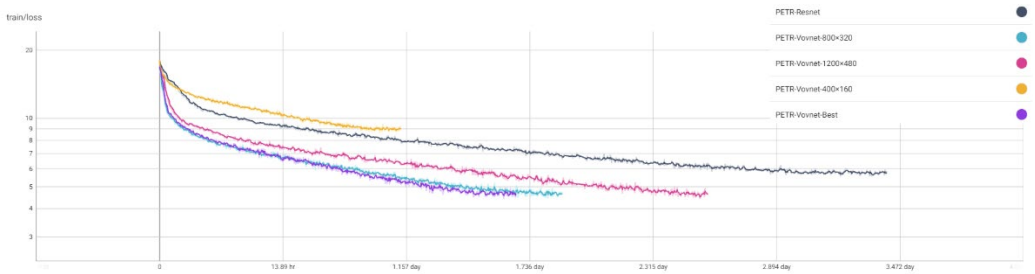

Figure 13: Training Loss Curve

In this experiment, the PETR algorithm possesses the optimal performance when using Vovnet as the backbone, with learning rate at 2e-5, optimizer set to Momentum Gradient Descent, gradient clipping parameter of 30, and an image input size of 800 x 320. Compared to the original result based on Resnet and PETR algorithm, the final optimized model improved its mAP from 0.3137 to 0.3794.

This performance enhancement can be attributed to three main factors:

1) The choice of backbone. Vovnet provides more comprehensive and specific feature information

ISSN 2616-5775 Vol. 7, Issue 8: 63-73, DOI: 10.25236/AJCIS.2024.070810

than Resnet. It applies OSA module to strengthen the collection of features inform from various neural layers. In contrast, Resnet adopts feature aggregation, which may lead to the loss of detailed features extracted by some neural layers. Hence, Vovnet is obviously superior to Resnet.

2) The image input size. The image size of 800 x 320, compared to 400 x 160, offers higher clarity and more detailed information. Compared to the image input size of 1200 x 480, 800 x 320 can avoid over-fitting that might occur due to the extraction of excessive detailed information from the larger image size. Therefore, the image input size of 800 x 320 has the optimal performance.

3) The modification of training parameters. By reducing learning rate and max norm, it can decrease the risk of over-fitting. Practice has proved that the modification can make the training proceed normally and effectively solve the impact of gradient explosion on model training[5-6].

During the model optimization process, there are some research challenges, mainly including:

1) High Time Consumption

In the process of model training, a NuScenes data set of approximately 300GB was used for training. The specification of dataset directly affects the training time, resulting in a relatively long training period. Additionally, more iterations are needed to achieve convergence due to the complexity of the model, thus increasing the training time.

2) Complex Environment Configuration

While creating an environment, it is necessary to consider the compatibility between different versions of dependent libraries. For example, when a specific library is compatible with other libraries of particular versions, the process of configuring the environment often encounters errors due to the various versions of systems and hardware involved.

5. Conclusions and Prospects

5.1. Conclusions

In this project, the application of deep learning methods to the problem of 3D object detection was explored. By using different feature extraction models and adjusting various model parameters, the original Resnet implementation of PETR algorithm has achieved an optimized effect on model performance. The PETR algorithm possesses the optimal performance when using Vovnet as the backbone, with learning rate at 2e-5, optimizer set to Momentum Gradient Descent, gradient clipping parameter of 30, and an image input size of 800 x 320. Compared to the original result based on Resnet and PETR algorithm, the final optimized model improved its mAP from 0.3137 to 0.3794, achieving higher accuracy and performance in target detection.

5.2. Prospects

1) Data augmentation techniques can be attempted to improve the model's robustness against factors such as illumination changes, occlusions, and background clutter. Original images from the dataset can be manipulated through rotation, cropping, and color adjustments to create fresh images. These newly generated images can then be integrated into the model's training regimen. Due to the diversity of training images, models can identify objects in various forms more effectively. For instance, while the original model could only recognize standing figures and failed to detect lying figures, the new model, by incorporating rotated standing images as training data, might be able to recognize lying figures, thereby improving the model's reliability.

2) Attempts can be made to use attention mechanisms to help the model focus on the parts of the image that are relevant to the target and suppress the impact of background clutter. By adding attention mechanisms to the backbone, the focus on effective targets can be increased at the feature extraction level, thereby enhancing the overall performance of the model.

3) By combing the optimized PETR algorithm with the object motion trend prediction system, the accurate positioning information of the object provided by the optimized PETR algorithm can be input into the object motion trend prediction system, thus enabling autonomous driving systems to anticipate potential hazards and address potential issues proactively.

ISSN 2616-5775 Vol. 7, Issue 8: 63-73, DOI: 10.25236/AJCIS.2024.070810

References

[1] Liu, Y., Wang, T., Zhang, X., & Sun, J. (2022, October). Petr: Position embedding transformation for multi-view 3d object detection. In European Conference on Computer Vision (pp. 531-548). Cham: Springer Nature Switzerland.

[2] Chen, Y., Liu, S., Shen, X., & Jia, J. (2020). Dsgn: Deep stereo geometry network for 3d object detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 12536-12545).

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

[4] Mousavian, A., Anguelov, D., Flynn, J., & Kosecka, J. (2017). 3d bounding box estimation using deep learning and geometry. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (pp. 7074-7082).

[5] Wang, X., Xiao, T., Jiang, Y., Shao, S., Sun, J., & Shen, C. (2018). Repulsion loss: Detecting pedestrians in a crowd. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7774-7783).

[6] Arnold, E., Al-Jarrah, O. Y., Dianati, M., Fallah, S., Oxtoby, D., & Mouzakitis, A. (2019). A survey on 3d object detection methods for autonomous driving applications. IEEE Transactions on Intelligent Transportation Systems, 20(10), 3782-3795.