Computer Modeling in
Engineering & Sciences

Tech Science Press

Doi:10.32604/cmes.2025.064775

ARTICLE

FastSECOND: Real-Time 3D Detection via Swin-Transformer Enhanced SECOND with Geometry-Aware Learning

Xinyu Li[1,2], **Gang Wan**[2], **Xinyang Chen**[3], **Liyue Qie**[3], **Xinnan Fan**[3], **Pengfei Shi**[3] and **Jin Wan**[3,*]

[1]College of Shipbuilding Engineering, Harbin Engineering University, Harbin, 150001, China
[2]China Yangtze River Electric Power Co., Ltd., Yichang, 443002, China
[3]College of Information Science and Engineering, Hohai University, Changzhou, 213000, China

*Corresponding Author: Jin Wan. Email: 230207040009@hhu.edu.cn

Received: 24 February 2025; Accepted: 01 July 2025; Published: 31 July 2025

ABSTRACT: The inherent limitations of 2D object detection, such as inadequate spatial reasoning and susceptibility to environmental occlusions, pose significant risks to the safety and reliability of autonomous driving systems. To address these challenges, this paper proposes an enhanced 3D object detection framework (FastSECOND) based on an optimized SECOND architecture, designed to achieve rapid and accurate perception in autonomous driving scenarios. Key innovations include: (1) Replacing the Rectified Linear Unit (ReLU) activation functions with the Gaussian Error Linear Unit (GELU) during voxel feature encoding and region proposal network stages, leveraging partial convolution to balance computational efficiency and detection accuracy; (2) Integrating a Swin-Transformer V2 module into the voxel backbone network to enhance feature extraction capabilities in sparse data; and (3) Introducing an optimized position regression loss combined with a geometry-aware Focal-EIoU loss function, which incorporates bounding box geometric correlations to accelerate network convergence. While this study currently focuses exclusively on the detection of the Car category, with experiments conducted on the Car class of the KITTI dataset, future work will extend to other categories such as Pedestrian and Cyclist to more comprehensively evaluate the generalization capability of the proposed framework. Extensive experimental results demonstrate that our framework achieves a more effective trade-off between detection accuracy and speed. Compared to the baseline SECOND model, it achieves a 21.9% relative improvement in 3D bounding box detection accuracy on the hard subset, while reducing inference time by 14 ms. These advancements underscore the framework's potential for enabling real-time, high-precision perception in autonomous driving applications.

KEYWORDS: 3D object detection; automatic driving; Deep Learning; SECOND; geometry-aware learning

1 Introduction

The evolution of deep learning has indeed led to significant advancements in two-dimensional object detection models, making them highly effective for various applications. However, the growing demands of the real world have revealed the limitations of two-dimensional object detection models. They are no longer adequate to handle the challenges posed by increasingly complex situations, especially those applications that require multi-dimensional information, such as autonomous driving, robotics, augmented reality, surveillance, and other cutting-edge domains [1].

To illustrate this point, consider the context of autonomous driving. Relying solely on two-dimensional object detection models in such scenarios can provide only a limited, flat view of the environment [2,3]. This inherent limitation substantially threatens the safety and the successful deployment of autonomous systems. In these advanced application contexts, there is a critical need to move beyond the constraints of two-dimensional detection. In the field of autonomous driving, existing two-dimensional object detection systems suffer from several limitations, such as the lack of depth information, poor performance in detecting occluded or overlapping objects, and a strong dependence on viewing angles. Therefore, three-dimensional object detection comes into being. In summary, while two-dimensional object detection models have made substantial progress, modern application scenarios' dynamic and intricate nature calls for more advanced solutions. Three-dimensional object detection has emerged as a vital area of research and development to address the heightened demands of safety and performance in domains like autonomous driving and beyond.

In recent years, computer vision has witnessed a remarkable evolution, marked by significant advancements in the perception and understanding of the three-dimensional (3D) world. Central to this progress is the domain of three-dimensional object detection, a critical area of research that plays a pivotal role in enabling machines to perceive and interact with their physical surroundings in a manner akin to human cognition. This emerging field addresses the fundamental challenge of locating and identifying objects within a 3D environment, providing a wealth of applications across various domains [4]. Consequently, three-dimensional object detection algorithms are gradually gaining attention. This model utilizes multiple sensors to obtain information about object categories, positions, orientations, sizes, and more [5]. In autonomous driving, three-dimensional point clouds are commonly acquired using LiDAR sensors [6]. Three-dimensional point clouds represent irregular and sparse data [7]. If the methods used in two-dimensional object detection are directly applied, depth information will be lost, resulting in reduced detection accuracy [8].

In response to the above challenge, Qi et al. introduced PointNet in 2017 [9], which addresses the issue of unordered and unaligned point clouds by applying symmetric functions and T-Net transformation matrices. This method gave rise to many PointNet-based techniques for processing raw point clouds, characterized by high computational complexity and accuracy. Drawing inspiration from the image pixel format, researchers proposed a voxel-based representation, leading to the development of voxelized detection networks based on VoxelNet [10]. The original VoxelNet employs 3D convolutions for voxelization, resulting in improved speed compared to point clouds, yet still time-consuming compared to two-dimensional object detection models. The SECOND network [11] introduced sparse convolutions as a substitute for 3D convolutions, thereby enhancing the efficiency of voxelization. The evolution of voxel design, progressing from manually engineered features to machine learning and ultimately incorporating sparse convolutions, has enabled simultaneous improvements in detection speed and accuracy. However, existing three-dimensional object detection systems still struggle to match the detection speed of their two-dimensional counterparts, which hinders their practical application in autonomous driving. Therefore, to enhance the applicability of 3D detection in this field, it is essential to improve the detection speed of 3D models while maintaining their detection accuracy.

In light of all this information, the novelty of the proposed FastSECOND can be given as follows:

1) Voxel-based 3D detection networks in previous studies typically rely on sparse convolutions for feature extraction. However, such convolutions have relatively fixed and localized receptive fields, which limit their ability to perform global modeling across voxels. In contrast, the Swin-Transformer v2 leverages a hierarchical window-based attention mechanism that effectively captures broader spatial relationships. Integrating the Swin-Transformer module into the sparse convolutional part of the model enables finer control over image details and contributes to the generation of more accurate features.

2) The activation function used in the network's Voxel Feature Encoding (VFE) module is traditionally the Rectified Linear Unit (ReLU) function. Although ReLU is simple and efficient, it tends to lose subtle yet

valuable features in sparse environments. Replacing ReLU with the Gaussian Error Linear Unit (GELU) activation function helps retain more informative signals, thereby enhancing the expressiveness and propagation of features in sparse point clouds, accelerating model convergence, and improving overall performance.

3) Previous studies often applied convolutions to all channels. In this study, considering the high similarity among channel-wise features, Channel-Partial Convolution (CPConv) is introduced in the Region Proposal Network (RPN) to reduce computational overhead. This method effectively lowers the computation cost while significantly enhancing detection speed.

4) Most existing research on voxel-based 3D detection networks still employs traditional IoU-based loss functions. However, these functions suffer from issues such as sample imbalance and insufficient utilization of the geometric information of bounding boxes. To address these limitations, we introduce the Focal-EIoU loss function to optimize the position regression loss. This function incorporates geometric factors during optimization, thereby improving regression accuracy.

2 Related Work

Lidar-based object detection methods have gained significant attention and popularity due to their high accuracy and efficiency. These methods utilize point clouds as their primary input data source. Point clouds contain valuable information such as the precise position, distance, depth, and angle of objects, which are often absent in two-dimensional images. This rich data composition provides a more faithful representation of real-world scenarios, particularly in complex environments like autonomous driving and robotics. However, it's essential to acknowledge the limitations of point clouds. They inherently possess the drawback of being sparse and irregular, which presents significant challenges in their processing and analysis. To overcome these challenges, researchers have developed three main approaches for point cloud processing: point-based Methods,view-based Methods, and voxel-based Methods.

2.1 Methods Based on Points

To tackle the inherent challenges posed by the unordered nature and rotation invariance of three-dimensional point clouds, Qi et al. introduced PointNet in 2017 [9]. PointNet addressed these issues by leveraging symmetric functions to process point clouds and introduced transformation matrices, referred to as T-Nets, to align input point clouds and features from different points. This groundbreaking work laid the foundation for handling unordered 3D data.

Building upon PointNet, PointNet++ [12] further improved the model's capabilities. It extended the concept by employing local point networks to process individual point sets, leading to the generation of local attribute features. Additionally, PointNet++ introduced hierarchical encoding, allowing for the extraction of more complex and informative features.

In 2019, Shi et al. introduced PointRCNN [13], a two-stage detection network tailored for 3D object detection. In the first stage, PointNet++ was employed for point cloud segmentation. In the second stage, fine-tuning was performed to refine the coordinates of each bounding box, ultimately enhancing detection accuracy. While effective, these methods primarily relied on PointNet for feature extraction, which led to substantial computational complexity, making real-time detection challenging.

One notable drawback of the two-stage networks mentioned above was the incorporation of upsampling layers for feature propagation, along with a refinement stage for fine-tuning candidate boxes. These additional steps significantly increased the computational time and complexified the network architecture.

In 2020, Yang et al. proposed a more lightweight 3D object detection model called 3DSSD [14]. This model eliminated the need for upsampling layers and refinement steps. Instead, during the downsampling process, 3DSSD introduced a fusion sampling strategy to detect a subset of representative points. This innovative approach streamlined the network structure, making it more computationally efficient and suitable for real-time applications with large-scale point clouds.

2.2 Methods Based on Views

View-based methods transform the three-dimensional problem into a two-dimensional image for processing. Subsequently, they predict three-dimensional information using two-dimensional object detection networks. In 2016, Li et al. introduced Velodyne Fully Convolutional Network (VeloFCN) [15], which employs cylindrical surface projection to project three-dimensional information onto a front view. Full convolutional operations are performed on the two-dimensional point map, and finally, upsampling is conducted for classification and regression, creating feature maps consistent with the original size. In 2018, Yang et al. proposed PIXOR (ORiented 3D object detection from PIXel-wise neural network predictions) [16], which transfers point cloud height information to color channels to achieve a top-down view representation. Subsequently, a two-dimensional detection network is employed for detection. In 2019, Simon et al. introduced Complex YOLO [17], a method that projects three-dimensional point clouds onto a two-dimensional top-down view. To address information loss resulting from point cloud projection, multiple features from the point cloud are combined to fill the input channels, compensating for the lost target information. However, due to the inherent loss of some three-dimensional information caused by projection, detection accuracy is generally lower for projection-based methods.

2.3 Methods Based on Voxels

Voxel-based methodologies represent a pivotal approach in processing three-dimensional point clouds, entailing the discretization of point clouds into three-dimensional voxels, with each voxel encapsulating pertinent points from the point cloud dataset. This technique has evolved significantly over time, with several seminal contributions in the field.

In the initial stages, voxel design relied on manual coding and voting schemes [18–20]. These early methods required meticulous manual crafting of voxel structures and subsequent voting mechanisms for object detection, which, while functional, lacked the automation and adaptability that later advancements would introduce.

In 2016, Zhou et al. presented a groundbreaking approach with VoxelNet [10]. VoxelNet marked a transformative shift by incorporating machine learning techniques to autonomously voxelate point clouds. A pivotal innovation introduced by VoxelNet was the voxel feature encoding layer. This layer was designed to encode intricate three-dimensional features within each voxel into vector representations. These voxel feature vectors were aggregated across multiple layers in intermediate convolutional networks. Subsequently, a Region Proposal Network (RPN) leveraged these aggregated feature vectors as input, yielding comprehensive three-dimensional detection results. VoxelNet was instrumental in automating voxelization and enhancing object detection accuracy through deep learning.

In 2018, Yan et al. introduced the SECOND network [11] as a significant evolution of VoxelNet. SECOND built upon the foundation laid by VoxelNet but introduced a novel approach to mitigate computational complexity. Notably, SECOND replaced conventional 3D convolutions with sparse convolutions, thereby reducing voxelization time and enhancing the efficiency of point cloud processing.

Building upon the principles of SECOND, Lang et al. introduced PointPillars [21] in 2019. PointPillars took a distinctive top-down perspective, dividing point clouds into pillars for two-dimensional feature extraction. This novel approach utilized two-dimensional convolutional networks and two-dimensional detection heads for object detection, showcasing a different paradigm within voxel-based methodologies.

In 2020, Shi et al. presented PV-RCNN [22], which harnessed three-dimensional sparse convolutions for point cloud encoding. PV-RCNN introduced innovative techniques, including voxel-to-key point scene coding and key-to-voxel Region of Interest (RoI) feature extraction operations. These techniques facilitated the aggregation of multi-scale semantic voxel features into key features through the voxel set aggregation module.

Crucially, the SECOND network, in particular, has emerged as a prominent and influential methodology in voxel-based object detection. Comprising three core components, namely the voxel feature extraction module, the sparse convolution module, and the RPN module, SECOND offers a holistic solution for three-dimensional object detection within point clouds. The voxel feature extraction module transforms raw point clouds into voxel representations, akin to VoxelNet. Meanwhile, the sparse convolution module, propelled by a GPU-based sparse convolution algorithm, adeptly handles the sparsity inherent in point cloud data. This module employs a sequence of six consecutive sparse convolutions followed by a sparse-to-dense layer, resulting in the transformation of three-dimensional sparse features into dense two-dimensional counterparts. Finally, the RPN module assumes the pivotal role of generating comprehensive results for the entire object detection network. The structure of SECOND is shown in Fig. 1.

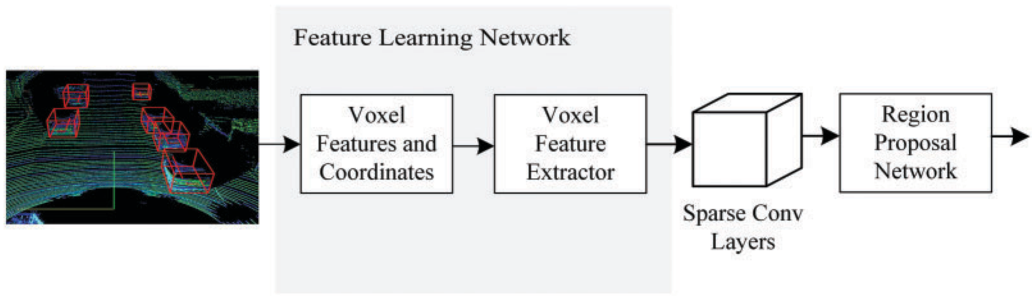

Figure 1: The structure of SECOND

This study aims to improve the SECOND network to enhance its overall performance in 3D object detection tasks. The primary challenge faced in the current research is how to improve detection accuracy while maintaining or further accelerating the network's inference speed. In other words, the improved model must not only outperform the original SECOND network in terms of accuracy, enabling more precise recognition and localization of objects, but also meet the real-time requirements for fast response in practical application scenarios. To address this, the study will focus on lightweight network design, efficient feature extraction mechanisms, and the introduction of multi-scale feature fusion strategies. The goal is to achieve an effective balance between detection accuracy and speed without significantly increasing computational overhead, providing a more efficient and reliable solution for 3D object detection systems.

3 Methodology

3.1 Region Proposed Network Based on Partial Convolution

The Region Proposal Network (RPN) within the SECOND framework, much like its counterpart in Faster R-CNN, plays a pivotal role in generating bounding boxes based on the feature maps extracted by the backbone network. However, it's essential to highlight a key distinction: SECOND does not incorporate the second stage found in Faster R-CNN, making it a unique approach in object detection for three-dimensional point clouds.

In the SECOND architecture, the RPN network takes the feature representations obtained from sparse Convolutional Neural Networks (CNNs) specifically tailored for processing three-dimensional point cloud data. These features are then cleverly transformed into a top-down view feature map. On this feature map, the RPN network performs two critical tasks. First, it determines the class of each proposed region within the map. Second, it calculates the regression values necessary to refine the bounding boxes associated with these regions. These tasks are accomplished using traditional fully convolutional networks, which, while effective, introduce a substantial computational burden.

Recognizing the computational complexity posed by these conventional fully convolutional networks, this study sets out with a clear objective: to enhance the RPN network's efficiency. The aim is to streamline the computational demands while maintaining or even improving the quality of region proposals. This optimization endeavor is crucial, especially in resource-constrained environments or scenarios where real-time performance is paramount.

Existing methods to reduce network computational complexity, such as MobileNet [23], ShuffleNet [24], and GhostNet [25], utilize depthwise convolutions (DWConv) and group convolutions (GConv) for feature extraction. Although these methods improve FLOPs efficiency, they suffer from low computation efficiency due to fragmented computations. Furthermore, these networks often involve additional data operations like concatenation, shuffling, and pooling, along with frequent memory access.

To address the above issue, this paper adopts Channel-Partial Convolution (CPConv), which differs from the concept of CPConv used in image inpainting, where convolution is applied to valid spatial regions of the image. In contrast, the CPConv in this study performs convolution only on a subset of feature channels. CPConv leverages the high similarity among feature maps across different channels; extracting features from all channels would introduce redundancy. Therefore, it applies standard convolution to a selected subset of input channels to extract spatial features, while leaving the remaining channels unchanged. Since CPConv operates on only part of the channels without affecting the others, its computational complexity is lower than that of conventional convolution methods.

We can compare CPConv with the more popular optimization convolution DWConv through calculation. For the input $I \in R^{c\times h\times w}$, DWConv applies c filters $W \in R^{k\times k}$ to calculate the output $O \in R^{c\times h\times w}$. Each filter slides over an input channel and contributes to an output channel. The FLOPs of DWConv is $h \times w \times k^2 \times c$ and the ordinary convolution floating-point operand. For CPConv, the first or last continuous channel c_p is calculated as a representative of the entire feature map, where the channel number ratio is $r = \frac{c_p}{c} = \frac{1}{4}$. So the FLOPs of PWConv is four times as many as CPConv, and the FLOPs of Conv is sixteen times as many as CPConv.

The fundamental principle of CPConv can be visually understood through the Fig. 2.

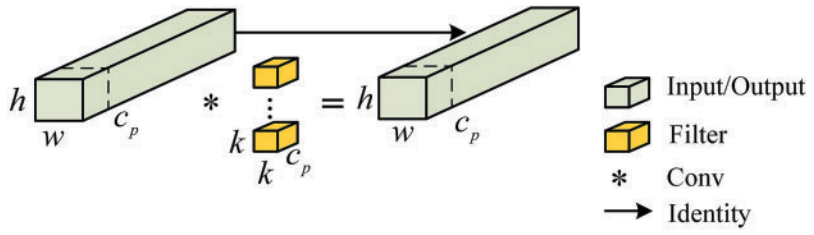

Figure 2: The principle of PConv

In this diagram, you can observe how CPConv selectively applies convolution to a specific portion of the input channels, preserving the valuable information contained within those channels. This technique optimizes the computational resources required for the convolutional process while retaining the richness of feature representation.

We replaced the normal convolutions with 3 × 3 CPConv and set the stride as 2. The improved network is shown in Fig. 3.

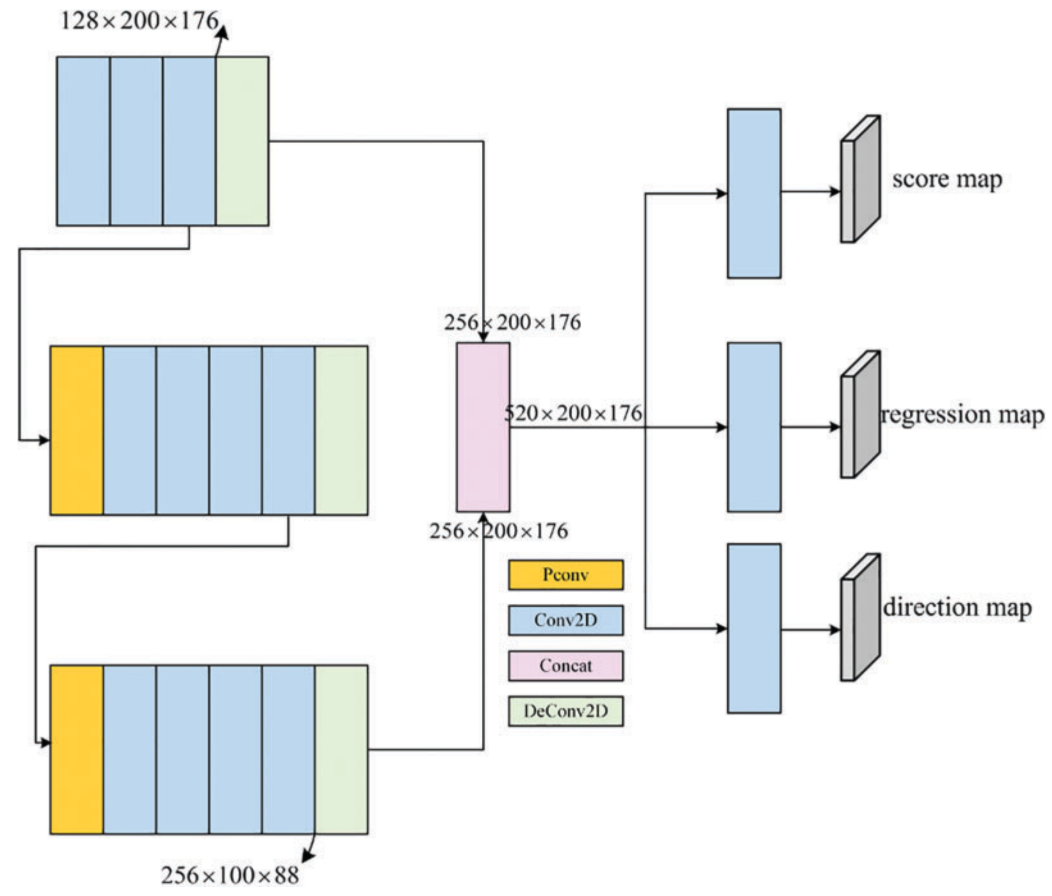

Figure 3: The improved RPN with CPConv

Furthermore, this study introduced modifications to the activation layers within the model. The traditional ReLU, while simple and efficient, has limitations in deep networks due to its hard thresholding at zero, which can lead to the loss of important information. To address this, the study adopts the GELU function, which offers several advantages.

The GELU function is a continuous and smooth non-linear activation function. The graph of the GELU function is visually represented in Fig. 4.

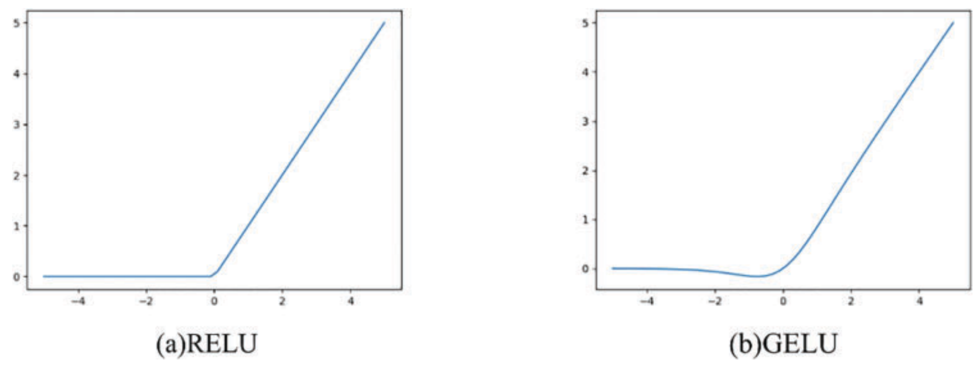

Figure 4: The function of RELU and GELU

As shown in the figure, the GELU function exhibits linear-like behavior near zero, resembling a linear function. However, as the input values increase, the function transitions into a linear saturation region. This smooth transition characteristic helps deep neural networks train more stably and allows the model to effectively handle a wide variety of input data types.

As a result, the GELU function contributes to faster convergence during the training process and improves overall performance. However, it is important to note that GELU comes with a relatively higher computational cost compared to ReLU. Therefore, to strike a balance between runtime efficiency and model performance, the study adopts GELU for the intermediate convolutional layers where its advantages are most pronounced, while retaining ReLU in other positions within the network architecture. This hybrid approach optimally combines the benefits of both activation functions, enhancing the overall efficiency and effectiveness of the model.

$$GELU(x) = 0.5x\left[1 + \tanh\left(\sqrt{\frac{2}{\pi}}\left(x + 0.044715x^3\right)\right)\right] \tag{1}$$

3.2 *Modification of Activation Functions in the VFE Layer*

The VFE (Voxel Feature Encoding) layer extracts voxel features. The processing of point cloud voxelization is shown in Fig. 5. In the original SECOND network, voxel-wise feature extraction is achieved by employing a fully connected network consisting of Batch Normalization (BN) layers and ReLU functions. Following this, voxel-wise max-pooling is utilized to gather locally aggregated features for each voxel. These locally aggregated voxel features are then flattened, and the resulting flattened local voxel features are combined with the features of each point. Subsequently, a Fully Convolutional Network (FCN) is employed to transform these per-point features into output features. This study incorporates a total of two VFE layers.

Expanding upon the advantages of GELU mentioned in subsection A, all ReLU functions in the second VFE layer are substituted with GELU functions. This replacement is aimed at harnessing the smoothness and faster convergence properties inherent in the GELU function, thereby enhancing the overall performance of the VFE section.

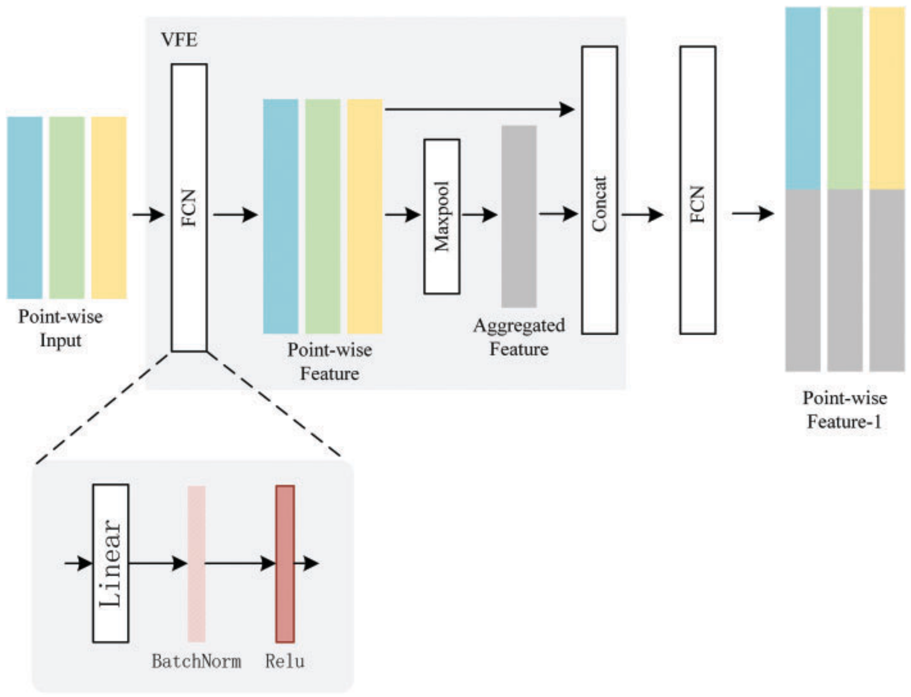

Figure 5: The processing of point cloud voxelization

3.3 Voxel Backbone Based on Swin-Transformer V2

To reduce the computational complexity of three-dimensional convolution operations, considering the complexity of vehicle distribution in autonomous driving scenarios, this chapter designs the backbone network part with the theory of Transformer. The Transformer model can capture global context information through its self-attention mechanism. This means that when processing images, the Transformer can consider information from all areas of the image, rather than just local areas. This is particularly important for understanding complex scenes and relationships in images.

Traditional Transformer utilizes multi-head self-attention modules to establish global information dependencies, enabling better extraction and analysis of global information. Swin Transformer is constructed based on shifting windows, as illustrated in Fig. 6. It consists of consecutive Swin-Transformer submodules. Each Swin-Transformer submodule includes a normalization layer, a multi-head self-attention module, a residual connection, and a two-layer MLP. Window-based Multi-head Self-Attention and Shifted Window Multi-head Self-Attention are used in two consecutive Swin Transformer submodules. Window Multi-heads Self-Attention increases the model's receptive field and decreases computational complexity by shifting features within local windows. Swin Transformer v2 [26] further improves upon the Swin Transformer with a more powerful self-attention mechanism. The main improvements are as follows: (1) To capture the global context information of images through the self-attention layer, the SCA (Sub Cosine Attention) formula is introduced, utilizing cosine similarity for computation as shown in Eq. (2). Here, $B \in R^{M^2 \times} M^2$ represents the relative positional bias parameter for each head, mitigating issues where attention in some modules and heads is dominated by a small number of pixels. This improvement corresponds to the red formula part in Fig. 6. (2) Further improvement in the design of the Swin Path includes Post Normalization, which normalizes the output of each residual module before merging it with the main branch. This enables more accurate control over the generation of image details, corresponding to the orange part in the figure. (3) Different from the original relative positional encoding bias, a logarithmic interval continuous positional bias is adopted to cope with significant changes in window size. Logarithmic scaling makes the differences minimal when window sizes change significantly, making the model more controllable, corresponding to the blue part in the figure. Here, Δx and Δy represent the coordinates of relative positional encoding, which are extrapolated through an MLP network to accommodate window size changes.

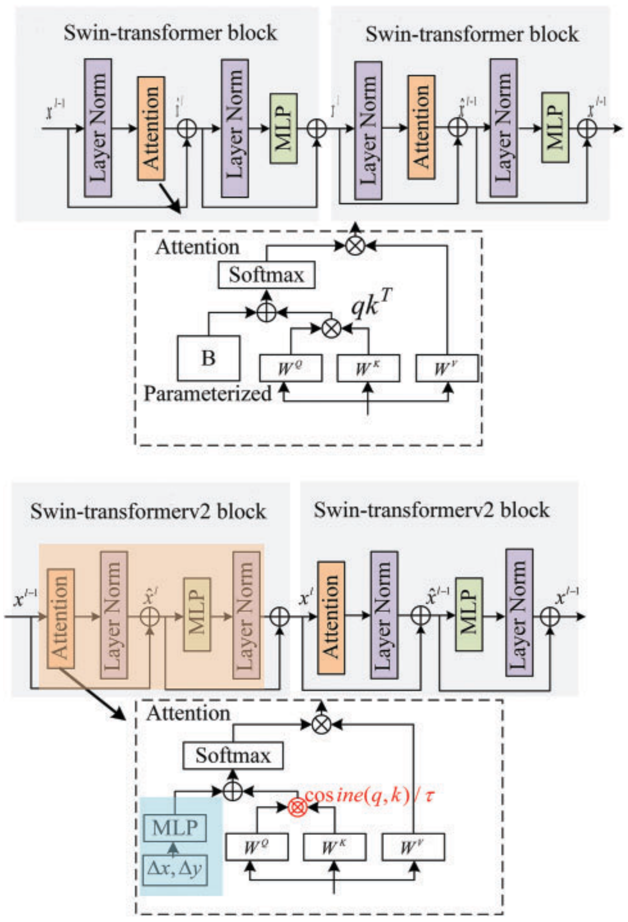

Figure 6: The principle of Swin-Transformer V2

Acknowledging the effective performance of the Swin-Transformer V2 module in the task of image feature extraction and its widespread adoption in large-scale vision models, this study seeks to leverage its capabilities by integrating the Swin-Transformer V2 module into the sparse convolution section of the architecture.

In the original SECOND network, the intermediate extraction module plays a pivotal role. This module combines submanifold convolution with sparse convolutions, resulting in the transformation of sparse tensors into denser data representations. Subsequently, these dense data representations are stacked along the Z-axis, effectively enhancing the feature extraction process.

By incorporating the Swin-Transformer V2 module into this intermediate extraction module, this study aims to harness the power of the Swin-Transformer V2 for extracting meaningful features from the sparse data inherent in 3D point cloud inputs. And we insert a Patch Merging layer before each Swin-Transformer V2 block due to the large amount of data processed by the sparse convolutional module.

The depth of the feature map is halved by subsampling and concatenation, which can improve the model performance and reduce the computing cost as much as possible. This integration represents a strategic enhancement that has the potential to boost the model's capability for capturing valuable spatial information, ultimately contributing to improved object detection performance in complex 3D environments. The modified structure is shown in Fig. 7.

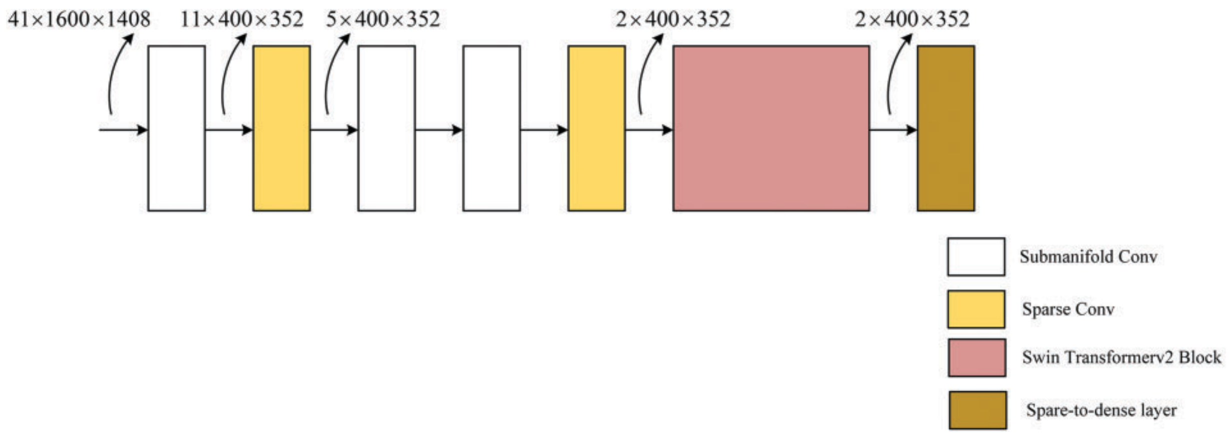

Figure 7: Voxel backbone based on swin-transformer v2

This design choice is grounded in the objective of enhancing the model's performance while simultaneously mitigating computational costs. By inserting the Swin-Transformer v2 blocks at this juncture in the network, the model can benefit from the capabilities of these blocks to capture and process intricate spatial and semantic information within the data. This enhancement is achieved without excessively increasing computational demands, as the number of Swin-Transformer v2 blocks and their associated computational load are carefully managed.

In essence, this strategic integration strikes a balance between model performance and computational efficiency, which is crucial in real-world applications, particularly in scenarios like autonomous driving where computational resources are often limited.

$$Sim(q_i, k_j) = \frac{cos(q_i, k_j)}{\tau} + B_{i,j} \tag{2}$$

3.4 Improvement of Loss Function

Due to the limitations of the original Intersection over Union (IoU) calculation method, which primarily assesses the overlap degree of bounding boxes without accurately considering positional information, this study adopts the Enhanced Intersection over Union (EIoU) Loss [27] as an alternative. The EIoU Loss is chosen to account for variations in the shape and size of bounding boxes, providing a more comprehensive and nuanced evaluation of bounding box similarity.

While the traditional IoU metric focuses on training the model based on the coordinate correlation between predicted and ground truth boxes, the EIoU Loss takes a different approach. It is designed to optimize for differences in aspect ratios between the boxes. To achieve this, the EIoU Loss disentangles the impact of aspect ratio by separately calculating the length and width factors for both the target and predicted boxes. This loss function consists of three components: overlap loss, center distance loss, and width-height loss. The first two components continue the approach in Complete Intersection over Union (CIoU) [28].

At the same time, the width-height loss directly minimizes the differences in width and height between the target and predicted boxes. This approach enhances convergence speed. The formula for EIoU Loss is as follows:

$$\begin{aligned} L_{EIoU} &= L_{IoU} + L_{dic} + L_{asp} \\ &= 1 - IoU + \frac{\rho^2(b, b^{gt})}{{c_\omega}^2 + {c_h}^2} + \frac{\rho^2(w, w^{gt})}{{c_\omega}^2} + \frac{\rho^2(h, h^{gt})}{{c_h}^2} \end{aligned} \tag{3}$$

where IoU is the intersection over the union between the predicted and ground truth boxes. b^t, b^{gt} is the Euclidean distance between the centers of the two boxes, represent the width and height of the minimum enclosing box that covers both the predicted and ground truth boxes, while denoting the width and height of the predicted and ground truth boxes, respectively.

Due to the differentiability of EIoU Loss, it can be directly employed in gradient descent optimization algorithms for training. This enables its integration with other loss functions and optimization methods. Therefore, the study combines FocalL1 Loss [29] with EIoU Loss to address the issue of imbalanced positive and negative samples in bounding box regression tasks. This integration reduces the optimization contribution of numerous anchor boxes with limited overlap with the target box during bounding box regression, thus focusing the regression process on high-quality anchor boxes.

The positional regression loss is then defined as follows, where is a hyperparameter controlling the curvature of the curve:

$$L_{reg-other} = L_{Focal-EIoU} = IoU^{\gamma} L_{EIoU} \tag{4}$$

The overall loss function consists of three components: classification loss, regression loss, and orientation loss:

$$L_{total} = \beta_1 L_{cls} + \beta_2 (L_{reg-\theta} + L_{reg-other}) + \beta_3 L_{dir} \tag{5}$$

L_{cls} is represented using Focal Loss as the classification loss. $L_{reg-\theta}$ is represented using a sine error based on SmoothL1 as the angle regression loss. $L_{reg-other}$ is represented using Focal-EIoU Loss as the position regression loss. L_{dir} is calculated using Softmax Loss for binary classification as the orientation regression loss. Following the original SECOND network, since using a smaller weight for the orientation loss can enhance the accuracy of recognizing the target direction, the loss function coefficients are set to $\beta_1 = 1.0$, $beta_2 = 1.0$, $beta_3 = 0.3$ for the three respective components.

The overall structure of the model is shown in Fig. 8.

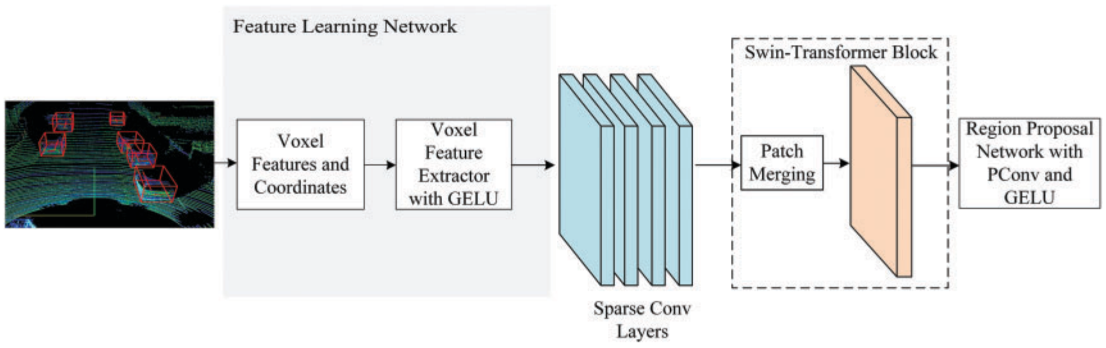
Feature Learning Network
Voxel Features and Coordinates
Voxel Feature Extractor with GELU
Sparse Conv Layers
Swin-Transformer Block
Patch Merging
Region Proposal Network with PConv and GELU

Figure 8: The structure of the modified SECOND

4 Experiments and Results

4.1 Dataset and Evaluation Metrics

In this study, the KITTI dataset serves as the primary dataset for evaluation. The KITTI dataset is a pioneering dataset designed for autonomous driving scenarios, comprising both LiDAR point clouds and RGB images captured by conventional cameras [14]. The 3D object detection portion of the KITTI dataset provides rich annotation information, including object categories, truncation levels, occlusion levels, observation angles, 2D bounding boxes, 3D dimensions, 3D positions, and rotation angles. Additionally, the dataset offers calibration information for the cameras, including projection matrices, rotation matrices, and the transformation matrix from LiDAR to camera. These details are crucial for projecting 3D object detection results onto the image plane. These parameters and variables make KITTI an effective resource for the development and evaluation of 3D object detection algorithms. The original KITTI training set comprises a total of 7481 samples, which are further split into 3712 training samples and 3769 validation samples. Additionally, the test set consists of 7518 samples. To provide a comprehensive evaluation, the KITTI dataset categorizes samples into three difficulty levels: easy, moderate, and hard, representing simple, normal, and complex samples, respectively.

The experimental platform setup for this study is as follows: The experiments are conducted on a system running Ubuntu version 22.04.4, with Python 3.7.13 and PyTorch 1.10.0 being utilized as the primary software tools. Model training is performed on four powerful 24GB NVIDIA GeForce RTX 3090 graphics cards. The ADAM optimizer is employed for model training, with an initial learning rate set to 0.003 and a batch size of 16 during training. Evaluation of detection accuracy, specifically Mean Average Precision (mAP), is carried out for both bird's-eye view and 3D boxes.

Given the complexity and volume of the KITTI dataset, which covers three distinct object classes (cars, pedestrians, and cyclists), the evaluation in this study is primarily focused on the Car class. Therefore, only the Mean Average Precision (mAP) for the Car class is calculated to assess the model's object detection performance on this specific class of objects. This approach streamlines the evaluation process and allows for a more targeted assessment of the model's effectiveness in detecting cars in various scenarios within the KITTI dataset.

To be specific, precision and recall are computed for the evaluation, with the formulas as follows:

$$Precision = \frac{TP}{TP + FP} \times 100\% \tag{6}$$

TP (True Positive) indicates the number of targets that are correctly classified, *FP* (False Positive) indicates the number of targets that are incorrectly detected when the background is the target. *AP* is the integral of the P-R curve. According to the truth value and the prediction box obtained, the function when the recall rate is r can be obtained. The AP formula is as follows:

$$AP = \int_0^1 p(r)dr \tag{7}$$

4.2 Contrast

To demonstrate the effectiveness of the model proposed in this paper, we compared ours with other established 3D object detection models, MV3D [19], F-PointNet [30], PointNet++ [12], PointRCNN [13], VoxelNet [10], 3DSSD [14], PointPillars [21], IA-SSD [31], PV-RCNN [22]and SECOND [11], as shown in Table 1. The comparisons are conducted on the KITTI Validation dataset. It is evident from the results that the proposed FastSECOND achieves a high detection accuracy while maintaining a relatively high detection speed. Notably, the proposed method exhibits superior performance on complex samples. In terms of detection time, the proposed FastSECOND is 14 ms faster than SECOND and approaches the speed of the lightweight network 3DSSD. Although IA-SSD is the fastest among all comparison models—being 4 ms faster per image than our model—our model achieves approximately 3% higher accuracy on the easy category and around 5% higher accuracy on the moderate and hard categories compared to IA-SSD. Specifically, in terms of 3D mAP, our model improves detection accuracy for hard samples by 14.5% compared to the original SECOND. In terms of Bird's Eye View mAP, our model achieves an 11.53% improvement in detection accuracy for hard samples over the original SECOND. Compared to PointPillars and PV-RCNN—both of which are enhanced versions of SECOND—our model attains similar accuracy while offering higher speed. Overall, our model achieves a better balance between detection accuracy and inference time, delivering higher accuracy with relatively less computational time. Table 1 shows the results.

Table 1: The accuracy of 3D object detection models on KITTI validation

Methods	**mTime (ms)**	**3D mAP (%)**			**Bird's Eye View mAP (%)**			**FPS**
		Easy	**Moderate**	**Hard**	**Easy**	**Moderate**	**Hard**	
MV3D	360	71.09	62.35	55.12	85.82	77	68.94	3
F-PointNet	170	81.2	69.79	60.59	88.7	84	75.33	5
PointNet++	200	83.44	71.11	65.74	89.19	85.64	76.71	5
PointRCNN	100	85.94	75.76	68.32	92.89	89.82	74.63	10
VoxelNet	230	77.47	65.11	57.73	89.35	79.29	77.39	4
3DSSD	38	87.73	78.58	72.01	91.99	85.64	80.45	26
PV-RCNN	91	92.57	84.83	82.69	95.76	91.11	88.93	11
IA-SSD	32	88.87	80.32	75.10	90.25	86.39	82.58	31
PointPillars	39	90.19	86.76	80.38	91.39	89.45	75.65	25
SECOND	50	83.13	73.66	66.2	88.28	84.2	76.73	20
VoxelNext	38	86.24	77.16	70.33	88.47	78.69	71.93	26
Ours	36	91.4	85.12	80.75	92.46	91.75	87.96	28

To further validate the effectiveness of the proposed FastSECOND, a comparison is made with the original unmodified SECOND. Visualizations of the predicted bounding boxes generated by both models are performed on KITTI Validation dataset point clouds. The specific comparative results are depicted in Fig. 9.

From the visualizations, it's evident that compared to the original SECOND, the proposed FastSECOND can identify vehicles that are partially occluded by buildings or other vehicles. Moreover, the proposed FastSECOND is effective in recognizing smaller distant objects. This observation underscores the effectiveness of the enhancements made to the RPN network, the intermediate extraction portion based on Swin-Transformerv2, the VFE, and the loss function in this study. These enhancements contribute to a significant improvement in vehicle detection accuracy while maintaining a high detection speed.

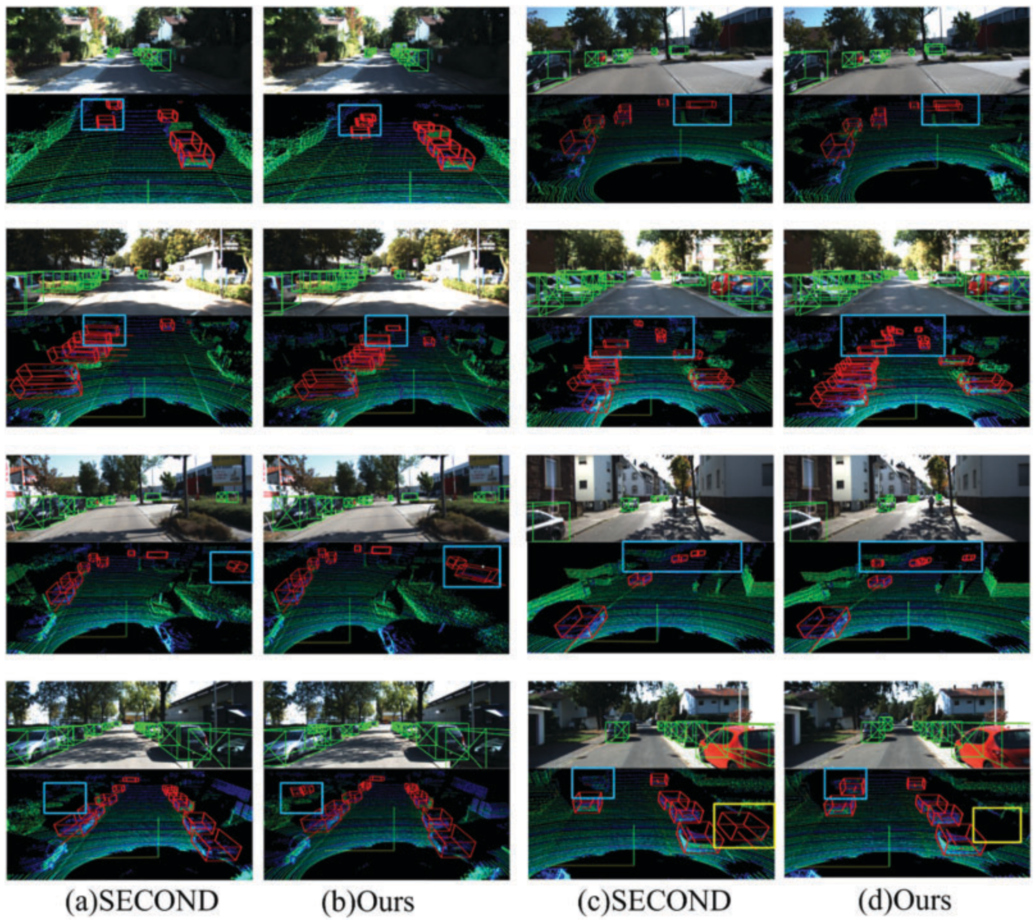

Figure 9: The visualization results on KITTI Validation (The blue boxes are the better parts of our work, and the yellow boxes are the detection errors of SECOND)

4.3 Ablation

4.3.1 Feasibility Analysis of GELU

To assess the feasibility of substituting ReLU with GELU in the VFE layer, a comprehensive series of experiments was meticulously designed to explore its impact. These experiments maintained all other variables constant and focused on three distinct scenarios:

1) employing ReLU in both VFE layers
2) adopting ReLU in one VFE layer while incorporating GELU in the other VFE layer.
3) employing GELU in both VFE layers. The outcomes of these experiments are thoughtfully presented in Table 2. Consequently, based on the experimental findings, this research paper opts for a configuration where one VFE layer employs GELU, while the other continues to use ReLU.

Table 2: Comparative results of different VFE layer activation functions

Number of layer used GeLU	mTimes (ms)	3D mAP (%)		
		Easy	Moderate	Hard
0	33 (±1.3)	88.98 (±0.93)	77.39 (±1.25)	71.54 (±1.38)
1	36 (±0.8)	91.49 (±0.62)	85.12 (±0.97)	80.55 (±1.12)
2	39 (±1.2)	91.92 (±0.57)	80.19 (±1.18)	74.02 (±1.23)

The transition to GELU functions yielded notable benefits, such as enhanced detection accuracy due to their faster convergence properties. However, it is important to acknowledge the trade-offs arising from the increased computational demands of GELU, which may lead to slower detection speed and other issues like overfitting or instability during the optimization process, resulting in reduced detection accuracy for both normal and complex samples. To ensure the reproducibility of results, the experiments in Table 2 were conducted using 5-fold cross-validation. Both the detection time and accuracy results presented in the table are averaged over the five folds, with the values in parentheses indicating the mean standard deviation. This approach enables a more robust evaluation by averaging results across different data splits, thereby reducing potential biases caused by random seed variations or error margins. By retaining GELU in only one VFE layer, we aim to strike a balance between improved convergence and computational efficiency, thus optimizing both detection accuracy and speed.

4.3.2 Impact of Various Improvements on Overall Results

To comprehensively evaluate the impact of various improvements on the overall performance, this study conducted ablation experiments, using the original SECOND model as a baseline. These experiments were conducted to dissect and understand the individual contributions of each enhancement. The result is shown in Table 3.

Table 3: Ablation studies for the coefficients of loss functions

Swin-Transformer V2 block	GeLU	CPConv	Times (ms)	3D AP (%)		
				Easy	Moderate	Hard
			50	83.13	73.66	66.20
✓			67	88.72	76.89	70.65
	✓		58	87.13	76.25	70.23
		✓	29	83.21	73.25	67.69
✓		✓	33	88.98	77.39	71.54
	✓	✓	32	87.37	76.51	71.23
✓	✓	✓	36	91.49	85.12	80.75

The experimental results demonstrate that the use of CPConv improves detection speed by approximately 40%, however, using CPConv alone does not lead to any improvement in detection accuracy. Regarding the Swin-Transformer V2 blocks and the use of the GeLU activation function, incorporating either one individually results in a modest accuracy improvement of about 5%. Nevertheless, when both are applied together, the detection accuracy increases by approximately 8% for easy samples, 12% for moderate samples, and 14% for hard samples. At the same time, detection speed improves by around 30% compared to the unmodified model. These results clearly demonstrate that the proposed enhancements in this study are both reasonable and effective.

In essence, the conducted experiments provide solid evidence that the proposed enhancements in this study not only effectively enhance the model's detection accuracy but also manage to maintain a reasonably fast detection speed. This suggests that the model achieves a well-balanced compromise between speed and precision, making it a promising approach to object detection.

4.3.3 The Coefficients of Loss Functions

Table 4 shows the ablation experiments for the coefficients of loss functions. The experimental results reveal that the model performs best when the coefficients β_1, β_2 and β_3 of loss functions are 1, 1 and 0.3, respectively. The combination of these coefficients yields the highest accuracy in terms of 3D mAP across all difficulty levels (easy, moderate, and hard). The model achieves an optimal balance between detection accuracy and computational efficiency, with the lowest mean time of 37.23 ms.

Table 4: The impact of each component improvement on the overall results

β_1	β_2	β_3	mTimes (ms)	3D mAP (%)		
				Easy	Moderate	Hard
0.5	0.5	0.3	36.94	90.45	85.67	77.92
1	1	0.3	37.23	90.41	86.12	79.71
0.5	1	0.3	37.56	89.32	84.56	77.55
1	0.5	0.3	37.45	91.12	86.00	79.31
0.5	0.5	0.4	37.78	88.76	86.32	79.15
1	1	0.5	37.34	89.90	85.56	78.80
0.5	1	0.6	37.65	89.23	86.10	78.94
1	0.5	0.7	37.01	89.95	85.99	79.25

4.3.4 Comparison of FLOPS among Different Convolution Methods

We conducted a comparative experiment on different convolution methods by analyzing their FLOPS. The experimental results are shown in Table 5.

Table 5: The FLOPS of different convolution methods

Conv	FLOPS (M)
Traditional Conv	97.281
DWConv	25.344
CPConv	6.152

The experimental results show that using PConv results in significantly lower computational complexity. Specifically, its computational complexity is 4.12 times lower than DWConv and 15.81 times lower than that of traditional convolution, which aligns well with the theoretical estimations.

4.3.5 The Impact of Channel Ratio r on the Model

For the selection of the channel ratio r in PConv, we determine an appropriate value by comparing the model's detection accuracy and computational complexity under different r settings. The experimental results are presented in Table 6.

Table 6: Comparative experiments of different channel ratios *r*

r	3D mAP (%)			FLOPS (M)
	Easy	Moderate	Hard	
1/2	91.57	85.23	80.62	13.241
1/4	91.49	85.12	80.75	6.152
1/8	86.21	79.23	75.14	3.214

Based on the comparative experimental results shown in the table, we can conclude that as the selected channel ratio increases, the model's detection accuracy improves. However, due to the high similarity among channel features, when the ratio reaches 1/4, further increasing the number of channels involved in convolution does not significantly enhance accuracy, but it does lead to higher computational complexity. Therefore, we selects a channel ratio of 1/4 as a compromise between detection accuracy and model complexity.

4.3.6 The Impact of Different Hardware on the Experimental Results

Considering the scalability and reproducibility of the model on different hardware, we conducted experiments on different GPU hardware. The experimental results are shown in Table 7.

Table 7: The impact of different GPUs on the overall results

GPU	3D mAP (%)			FPS
	Easy	Moderate	Hard	
2080	89.43	84.53	79.87	20
3090	91.49	85.26	80.75	28

The results show that our model can be successfully reproduced on lower-performance GPUs such as the 2080, with detection accuracy remaining virtually unaffected. However, due to hardware limitations, the detection speed decreases on lower-end GPUs.

5 Conclusion

This paper presents an improved 3D object detection model, built on the SECOND architecture, aimed at enhancing detection accuracy while maintaining its original advantage of fast detection. The proposed FastSECOND model is evaluated on the KITTI dataset, focusing primarily on the Car category. Experimental results show a significant improvement in detection accuracy, with the model achieving a maximum relative percentage improvement of 21.9% in the 3D hard detection task compared to the SECOND model, while detection time is reduced by 14 milliseconds. However, it should be noted that the current evaluation is limited to the Car category. Future work will expand to other categories, such as Pedestrian and Cyclist, to more comprehensively assess the generalization ability of this framework.

Acknowledgement: None.

Funding Statement: This research was funded by the National Key R&D Program of China (Grant No. 2022YFB4703400), the China Three Gorges Corporation (Grant No. 2324020012), the National Natural Science Foundation of China (Grant No. 62476080), the Jiangsu Province Natural Science Foundation (Grant No. BK20231186) and Key Laboratory about Maritime Intelligent Network Information Technology of the Ministry of Education (EKLMIC202405).

Author Contributions: Conceptualization, Xinyu Li and Gang Wan; methodology, Xinyu Li and Gang Wan; software, Xinyang Chen; validation, Xinyu Li, Xinyang Chen, Gang Wan and Xinnan Fan; formal analysis, Liyue Qie; investigation, Jin Wan; resources, Xinyu Li and Gang Wan; data curation, Xinyu Li and Jin Wan; writing—original draft preparation, Gang Wan; writing—review and editing, Xinnan Fan; visualization, Jin Wan and Pengfei Shi; supervision, Xinyang Chen and Xinnan Fan; project administration, Xinnan Fan and Gang Wan; funding acquisition, Xinnan Fan and Pengfei Shi. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data presented in this study are available on request from the corresponding author.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Zaidi SSA, Ansari MS, Aslam A, Kanwal N, Asghar M, Lee B. A survey of modern deep learning based object detection models. Digit Signal Process. 2022;126(11):103514. doi:10.1016/j.dsp.2022.103514.
2. Mao J, Niu M, Jiang C, Liang H, Chen J, Liang X, et al. One million scenes for autonomous driving: once dataset. arXiv:2106.11037. 2021.
3. Zhang R, Mao J, Wang H, Li B, Cheng X, Yang L. A survey on federated learning in intelligent transportation systems. IEEE Trans Intell Vehicles. 2024;1–17. doi:10.1109/TIV.2024.3446319.
4. Wei Z, Zhang F, Chang S, Liu Y, Wu H, Feng Z. Mmwave radar and vision fusion for object detection in autonomous driving: a review. Sensors. 2022;22(7):2542. doi:10.3390/s22072542.
5. Wang L, Bai Z, Zhou M, Ren X, Wang Y, Zhao X. Multimodal 3D target detection based on improved Mvxnet. In: 2024 9th International Conference on Image, Vision and Computing (ICIVC); 2024 Jul 15–17; Suzhou, China. p. 128–34.
6. Cunha L, Roriz R, Pinto S, Gomes T. Hardware-accelerated data decoding and reconstruction for automotive LiDAR sensors. IEEE Transact Vehic Technolo. 2023;72(4):4267–76. doi:10.1109/tvt.2022.3223231.
7. Jiang H, Lu Y, Zhang D, Shi Y, Wang J. Deep learning-based fusion networks with high-order attention mechanism for 3D object detection in autonomous driving scenarios. Appl Soft Comput. 2024;152(19):111253. doi:10.1016/j.asoc.2024.111253.
8. Pan X, Xia Z, Song S, Li LE, Huang G. 3D object detection with pointformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2021 Jun 20–25; Nashville, TN, USA. p. 7463–72.
9. Qi CR, Su H, Mo K, Guibas LJ. Pointnet: deep learning on point sets for 3D classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2017 Jul 21–26; Honolulu, HI, USA. p. 652–60.
10. Zhou Y, Tuzel O. Voxelnet: end-to-end learning for point cloud based 3D object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018 Jun 18–23; Salt Lake City, UT, USA. p. 4490–9.
11. Yan Y, Mao Y, Li B. Second: sparsely embedded convolutional detection. Sensors. 2018;18(10):3337. doi:10.3390/s18103337.

12. Qian G, Li Y, Peng H, Mai J, Hammoud H, Elhoseiny M, et al. Pointnext: revisiting pointnet++ with improved training and scaling strategies. Adv Neu Inform Process Syst. 2022;35:23192–204.
13. Shi S, Wang X, Li H. Pointrcnn: 3D object proposal generation and detection from point cloud. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2019 Jun 15–20; Long Beach, CA, USA. p. 770–9.
14. Yang Z, Sun Y, Liu S, Jia J. 3DSSD: point-based 3D single stage object detector. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020 Jun 13–19; Seattle, WA, USA. p. 11040–8.
15. Li B, Zhang T, Xia T. Vehicle detection from 3D lidar using fully convolutional network. arXiv:1608.07916. 2016.
16. Yang B, Luo W, Urtasun R. Pixor: real-time 3D object detection from point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018 Jun 18–23; Salt Lake City, UT, USA. p. 7652–60.
17. Simon M, Milz S, Amende K, Gross HM. Complex-yolo: real-time 3D object detection on point clouds. arXiv:1803.06199. 2018.
18. Zhao L, Guo J, Xu D, Sheng L. Transformer3D-Det: improving 3D object detection by vote refinement. IEEE Transact Circ Syst Video Technol. 2021;31(12):4735–46. doi:10.1109/tcsvt.2021.3102025.
19. Wang Y, Mao Q, Zhu H, Deng J, Zhang Y, Ji J, et al. Multi-modal 3D object detection in autonomous driving: a survey. Int J Comput Vis. 2023;131(8):2122–52. doi:10.1007/s11263-023-01784-z.
20. Yang W, Li Z, Wang C, Li J. A multi-task Faster R-CNN method for 3D vehicle detection based on a single image. Appl Soft Comput. 2020;95(2):106533. doi:10.1016/j.asoc.2020.106533.
21. Lang AH, Vora S, Caesar H, Zhou L, Yang J, Beijbom O. Pointpillars: fast encoders for object detection from point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2019 Jun 15–20; Long Beach, CA, USA. p. 12697–705.
22. Shi S, Guo C, Jiang L, Wang Z, Shi J, Wang X, et al. PV-RCNN: point-voxel feature set abstraction for 3D object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020 Jun 13–19; Seattle, WA, USA. p. 10529–38.
23. Howard A, Sandler M, Chu G, Chen LC, Chen B, Tan M, et al. Searching for mobilenetv3. In: Proceedings of the IEEE/CVF International Conference on Computer Vision; 2019 Oct 27–Nov 2; Seoul, Republic of Korea. p. 1314–24.
24. Zhang X, Zhou X, Lin M, Sun J. Shufflenet: an extremely efficient convolutional neural network for mobile devices. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018 Jun 18–23; Salt Lake City, UT, USA. p. 6848–56.
25. Han K, Wang Y, Tian Q, Guo J, Xu C, Xu C. Ghostnet: more features from cheap operations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2020 Jun 13–19; Seattle, WA, USA. p. 1580–9.
26. Liu Z, Hu H, Lin Y, Yao Z, Xie Z, Wei Y, et al. Swin transformer v2: scaling up capacity and resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2022 Jun 18–24; New Orleans, LA, USA. p. 12009–19.
27. Yang Z, Wang X, Li J. EIoU: an improved vehicle detection algorithm based on vehiclenet neural network. J Phys Conf Ser. 1924:012001. doi:10.1088/1742-6596/1924/1/012001.
28. Zheng Z, Wang P, Liu W, Li J, Ye R, Ren D. Distance-IoU loss: faster and better learning for bounding box regression. Proc AAAI Conf Artif Intell. 2020;34(7):12993–3000. doi:10.1609/aaai.v34i07.6999.
29. Hossain MS, Betts JM, Paplinski AP. Dual Focal Loss to address class imbalance in semantic segmentation. Neurocomputing. 2021;462(28):69–87. doi:10.1016/j.neucom.2021.07.055.
30. Qi CR, Liu W, Wu C, Su H, Guibas LJ. Frustum pointnets for 3D object detection from RGB-D data. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018 Jun 18–23; Salt Lake City, UT, USA. p. 918–27.
31. Zhang Y, Hu Q, Xu G, Ma Y, Wan J, Guo Y. Not all points are equal: learning highly efficient point-based detectors for 3D lidar point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2022 Jun 18–24; New Orleans, LA, USA. p. 18953–62.