applied sciences

MDPI

Article

A Lightweight Model for 3D Point Cloud Object Detection

Ziyi Li [1], Yang Li [2], Yanping Wang [2,*], Guangda Xie [1], Hongquan Qu [2] and Zhuoyang Lyu [3]

[1] College of Electrical and Control Engineering, North China University of Technology, Beijing 100144, China; lzy-11@mail.ncut.edu.cn (Z.L.); xgd_hhh@mail.ncut.edu.cn (G.X.)
[2] College of Information, North China University of Technology, Beijing 100144, China; liyang_ncut@ncut.edu.cn (Y.L.); qhqphd@ncut.edu.cn (H.Q.)
[3] College of Computer Science and Applied Math, Brown University, Providence, RI 02912, USA; zhuoyang_lyu@brown.edu
* Correspondence: wangyp@ncut.edu.cn

Abstract: With the rapid development of deep learning, more and more complex models are applied to 3D point cloud object detection to improve accuracy. In general, the more complex the model, the better the performance and the greater the computational resource consumption it has. However, complex models are incompatible for deployment on edge devices with restricted memory, so accurate and efficient 3D point cloud object detection processing is necessary. Recently, a lightweight model design has been proposed as one type of effective model compression that aims to design more efficient network computing methods. In this paper, a lightweight 3D point cloud object detection network architecture is proposed. The core innovation of the proposal consists of a lightweight 3D sparse convolution layer module (LW-Sconv module) and knowledge distillation loss. Firstly, in the LW-Sconv module, factorized convolution and group convolution are applied to the standard 3D sparse convolution layer. As the basic component of the lightweight 3D point cloud object detection network proposed in this paper, the LW-Sconv module greatly reduces the complexity of the network. Then, the knowledge distillation loss is used to guide the training of the lightweight network proposed in this paper to further improve the detection accuracy. Finally, extensive experiments are performed to verify the algorithm proposed in this paper. Compared with the baseline model, the proposed model can reduce the FLOPs and parameters by 3.7 times and 7.9 times, respectively. The lightweight model trained with knowledge distillation loss achieves comparable accuracy to the baseline. Experiments show that the proposed method greatly reduces the model complexity while ensuring detection accuracy.

Keywords: lightweight model design; 3D sparse convolution; object detection

check for updates

Citation: Li, Z.; Li, Y.; Wang, Y.; Xie, G.; Qu, H.; Lyu, Z. A Lightweight Model for 3D Point Cloud Object Detection. *Appl. Sci.* **2023**, *13*, 6754. https://doi.org/10.3390/app13116754

Academic Editor: Atsushi Mase

Received: 9 May 2023
Revised: 30 May 2023
Accepted: 31 May 2023
Published: 1 June 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Recently, autonomous driving has attracted more and more attention. As an important part of the self-driving vehicle perception system, 3D object detection can intelligently predict the location, size, and category of key 3D objects near the vehicle and provide accurate environmental information for the vehicle. Light detection and ranging (LiDAR)-based methods [1] occupy a major position in the field of 3D object detection. However, as the precision of the scanning equipment continues to increase, the scale of the original point cloud data becomes huge. Advanced 3D detectors are often accompanied by complex network structures that require billions of floating point operations (FLOPs) and are unable to be deployed on computationally limited platforms. Sufficient model compression techniques have been proposed in computer vision tasks to solve this problem, such as network pruning [2–4], quantization [5–7], lightweight model design [8–10], and knowledge distillation [11–13]. However, there are few studies on model compression in the 3D field. This paper aims to explore an efficient 3D convolutional neural network architecture that can meet the computing range required by our edge devices.

It is well known that the convolution layer is the most important 'time killer' in neural networks. In the above model compression technology, lightweight model design is used to improve the calculation way of the convolution layer, aiming at reducing the complexity of the model and achieving a more efficient network structure. We note the advanced lightweight models in the two-dimensional field. For example, Refs. [14–16] apply grouped point-wise convolution for the input data, reducing the dimension of the data and the number of parameters of the convolution layer; Refs. [17–19] use factorized convolution to design the lightweight model, which can effectively extract features and reduce the computational complexity of the model. Based on the above two ideas, this paper designs a novel 3D sparse convolution layer module (LW-Sconv module) as the basic module of an efficient 3D object detection network. The core part of the module consists of point-wise 3D convolution and depth-wise 3D sparse convolution. Simultaneously, transpose and reshape operations are introduced to help information flow between channels.

In this paper, an effective lightweight 3D point cloud object detection algorithm is proposed. The framework of the algorithm is composed of a novel 3D sparse convolution layer module. In addition, the algorithm is supervised and trained by three-part knowledge distillation loss, which effectively improves the detection accuracy of the algorithm. Experimental results on two public datasets show that the required operands and parameters are significantly reduced while ensuring accuracy. The main contributions of the method proposed in this paper are summarized as follows:

(1) A novel 3D sparse convolution layer module. In order to be able to deploy the model on memory-limited devices, we design an LW-Sconv module using factorized convolution and group convolution to replace the 3D sparse convolutional layers. The aim is to reduce the complexity of the model.
(2) An effective lightweight 3D object detector. Through the joint learning method, the detector is trained through three-part knowledge transfer, i.e., relation transfer, feature transfer, and output transfer. It is effective in obtaining detectors with the best detection performance.

The rest of this paper is organized as follows: The related work is reviewed in Section 2. Then, the detailed framework of the algorithm is described in Section 3 and the algorithm is evaluated by experiments in Section 4. Section 5 ends with a summary and conclusion.

2. Related Work

2.1. Lightweight Model Design

The amount of calculation and parameters of the 3D point cloud object detection algorithm based on convolutional neural networks are mainly determined by the convolution layer and the fully connected layer. Therefore, most of the existing model acceleration methods are considered from the perspective of reducing the computational complexity of the convolution process. These methods redesign a network structure with lower computational overhead and space memory consumption. Ref. [20] proposes a new network structure that increases the nonlinearity of the network and reduces the complexity of the model by adding an additional layer of 1×1 convolution. In order to reduce the storage requirements of the CNN model, it also removes the fully connected layer of the network and uses a global average pool. Group convolution [21] is another commonly used strategy to reduce the amount of network computation. GoogLeNet [14] uses a large number of group convolutions and the effectiveness of group convolution is verified by combining different convolution kernels. Ref. [16] proposes that, by using a large number of 1×1 convolution and group convolution strategies, the compression of about 50 times the parameters of AlexNet [22] is finally achieved without reducing the accuracy; the model is named SqueezeNet. RestNeXt [19] uses group convolution and the performance exceeds RestNet [23] when it has the same order of magnitude of parameters and FLOPs. In MobileNet [17], deep separable convolution is proposed. It replaces the standard convolution with a deep-wise convolution layer and a point-wise convolution layer. MobileNet can be faster than the VGG16 [24] network. ShuffleNet [15] introduces the concept of channel

shuffle. By cross-mixing the channels between different group convolutions, the feature information learned by each group convolution can be exchanged. In the last two years, Ref. [25] has used dropout to design a lightweight CNN network to reduce the number of parameters and Ref. [26] designed a lightweight model based on a novel convolutional block to prevent overfitting. The block takes advantage of separable convolution and squeeze–expand operations.

2.2. Three-Dimensional Object Detection

The development of deep learning has promoted the research of 3D point cloud object detection and the construction of deeper and larger convolutional neural networks has become the mainstream trend. Representative methods are: PointNet [27], which uses a multilayer perceptron to learn the spatial features of points and maximum pooling to aggregate global features; VoxelNet [28], which divides the point cloud into multiple voxels, characterizes the point cloud data by voxel feature encoding, and extracts features by convolutional middle layer; PointPillars [29], which, unlike VoxelNet, divides the point cloud into different vertical cylinders and projects the point cloud in each cylinder onto a 2D feature map; PointRCNN [30], which uses a deformation-based convolution for feature extraction, retaining local structure information; GLENet [31], which constructs probabilistic detectors by generating uncertainty labels and proposes an uncertainty-aware quality estimator architecture to guide the training of IoU branches with predictive localization uncertainty.

However, these advanced 3D detection models have high complexity and are unable to be deployed in real-time applications such as autonomous driving. Exploring an efficient 3D object detection model has become a research hotspot. Among them, SECOND [32] uses 3D sparse convolution instead of 3D standard convolution. A structured knowledge distillation framework is proposed in PointDistiller [33] to obtain a lightweight 3D point cloud object detection model. Ref. [34] proposes to simplify KNN search and graph shuffling to improve the efficiency of convolution. Ref. [35] combines the features of the point-based branch and the voxel-based branch, which not only perform effective feature extraction but also reduce memory occupancy. Ref. [36] uses sparse point-voxel convolution instead of point-voxel convolution to reduce the complexity of the model. For the indoor 3D target detection task, Ref. [37] proposes a generative sparse detection network, where the key component of the model is a generative sparse tensor decoder that uses a series of transposed convolution and pruning layers to extend the support of sparse tensors while discarding unlikely object centers to maintain minimal runtime and memory footprint. Ref. [38] proposes an anchor-free method to achieve 3D target detection using a purely data-driven approach. A novel oriented bounding box parameterization is also introduced that reduces the number of hyperparameters.

3. Method

This section provides a detailed description of the lightweight 3D point cloud object detection algorithm proposed in this paper. First, the LW-Sconv module is introduced, which is the basic block for constructing the network structure. Then, the knowledge distillation loss used to improve the performance of the algorithm is described.

3.1. Lightweight 3D Sparse Convolution Layer Module

The core part of LW-Sconv module consists of a group convolution, a transpose and reshape operation, and a depth-wise sparse convolution. The filter of the group convolution is $1 \times 1 \times 1$ and the filter of the depth-wise sparse convolution is $3 \times 3 \times 3$. Each convolution operation is followed by batch normalization and ReLU. Due to the 3D sparse convolution having two types (sub-manifold sparse convolution and regular sparse convolution), we describe the two types of the LW-Sconv module in Figure 1. There are three adjustable hyperparameters in the LW-Sconv module: g, C, and C'. g represents the number of groups in the $1 \times 1 \times 1$ group convolution layer; C represents the number of filters in the group convolution layer; C' represents the number of filters in the depth-wise sparse convolution

layer. This paper sets $C < C'$, which limits the number of input channels of depth-wise sparse convolution.

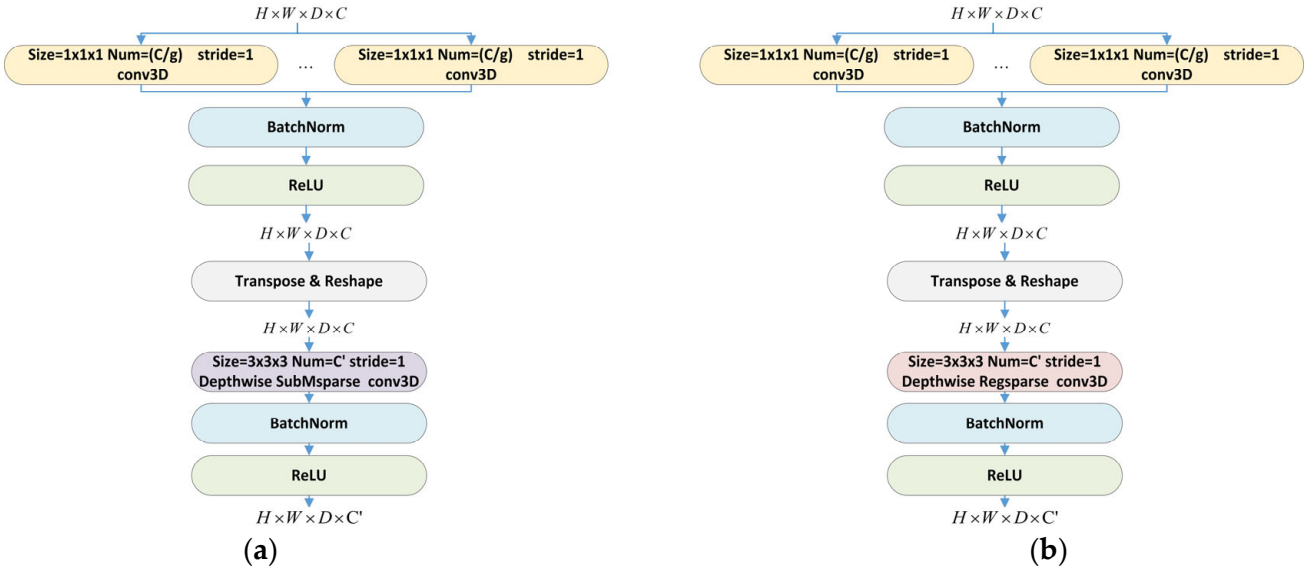


Figure 1. The architecture of the LW-Sconv module. (**a**) A lightweight sub-manifold sparse layer module (LW-Subsconv module). (**b**) A lightweight regular sparse layer module (LW-Regsconv module). $H \times W \times D \times C$ represents the input size. $H \times W \times D \times C'$ represents the output size. Size represents the size of the convolution kernel. Num represents the number of convolution kernels.

The overall goal of the lightweight model design in this paper is to change the calculation method of standard 3D sparse convolution and to identify a convolution neural network architecture with lower complexity. Therefore, this paper designs a lightweight 3D sparse convolution layer module with the following basis:

- Using $1 \times 1 \times 1$ group convolution to process the input data, not only are the parameters of the filter reduced compared with the ordinary $3 \times 3 \times 3$ filter, but the $1 \times 1 \times 1$ convolution after grouping is more suitable for lightweight networks with constrained complexity than the dense $1 \times 1 \times 1$ convolution;
- Applying transpose and reshape operations, the feature map is concatenated from the output of different group convolutions on the previous layer. The application of transpose and reshape operations is to avoid the non-circulation of information caused by grouping, which is conducive to the subsequent extraction of global features;
- Applying $3 \times 3 \times 3$ depth-wise sparse convolution, consider a standard 3D convolution, where the number of parameters in this layer is (the number of input channel) $\times$ (the number of filter) $\times$ $(3 \times 3 \times 3)$. Using the depth-wise sparse convolution, the feature map is convolved channel-by-channel, which maintains a small number of parameters. Compared with a $1 \times 1 \times 1$ filter, the receptive field becomes larger, the information read increases, and the global features obtained are better.

The above lightweight sparse convolution layer module design applies the ideas of group convolution and depth-wise convolution to sparse convolution, which can be achieved by highly optimized general matrix multiplication (GEMM). Sparse convolution is similar to the standard convolution calculation process, which uses GEMM to accelerate matrix operations. Therefore, the above lightweight method is also applicable and effective in sparse convolution. The difference is that the standard convolution uses im2col to gather and scatter, while the sparse convolution is based on the rulebook and hash table constructed in advance to construct the matrix and restore the spatial position. Due to the fact that the sparse convolution is different from the standard convolution in the way of matrix operation through GEMM, only the non-empty features in the feature map are stored when constructing the rulebook, so the feature map of size $H \times W \times D \times C$

is converted to $N \times C$, where N represents the number of non-empty positions and C represents the feature dimension. Using the formula of FLOPs in [17], the FLOPs of the standard sparse convolution layer are $3 \times 3 \times 3 \times C \times C'$. The FLOPs of the LW-Sconv module are $N \times C + 3 \times 3 \times 3 \times N \times C'$. It can be seen that the FLOPs have been greatly reduced after the lightweight design.

3.2. Lightweight 3D Point Cloud Object Detection Algorithm Framework

The overall framework of the algorithm in this paper is described in this section; the detailed network architecture is illustrated in Figure 2. Figure 2a shows the workflow of the algorithm. Firstly, the raw point cloud performs the voxelization to obtain voxels, which are used as the input of the network. The input voxels are encoded by voxel feature encoding (VFE) [28] to obtain voxel feature mapping. Then, the voxel features are sent to the 3D backbone. The extracted 3D features reshape to bird's eye view (BEV) along the z-axis direction and are fed into the 2D backbone for further feature extraction. Finally, they are sent to the detect head to obtain the output of the network. The most important feature is the architecture of the backbone, which is shown in Figure 2b. The 3D backbone in the algorithm is composed of 11 LW-Sconv modules and the 2D backbone is composed of 12 lightweight convolution layer modules (LW-conv module). For the 2D backbone, the design idea of the lightweight convolution layer module is the same as that of the 3D, but the filter is 2D. For the 3D backbone, an LW-Sconv module is not directly applied at the beginning because the number of input channels is relatively small. Starting from the second layer, the LW-Subsconv module and the LW-Regsconv module are stacked. The first convolution layer module in the second and third stage uses stride = 2. Other hyperparameters remain unchanged in the same stage and the number of output channels in the next stage is doubled.

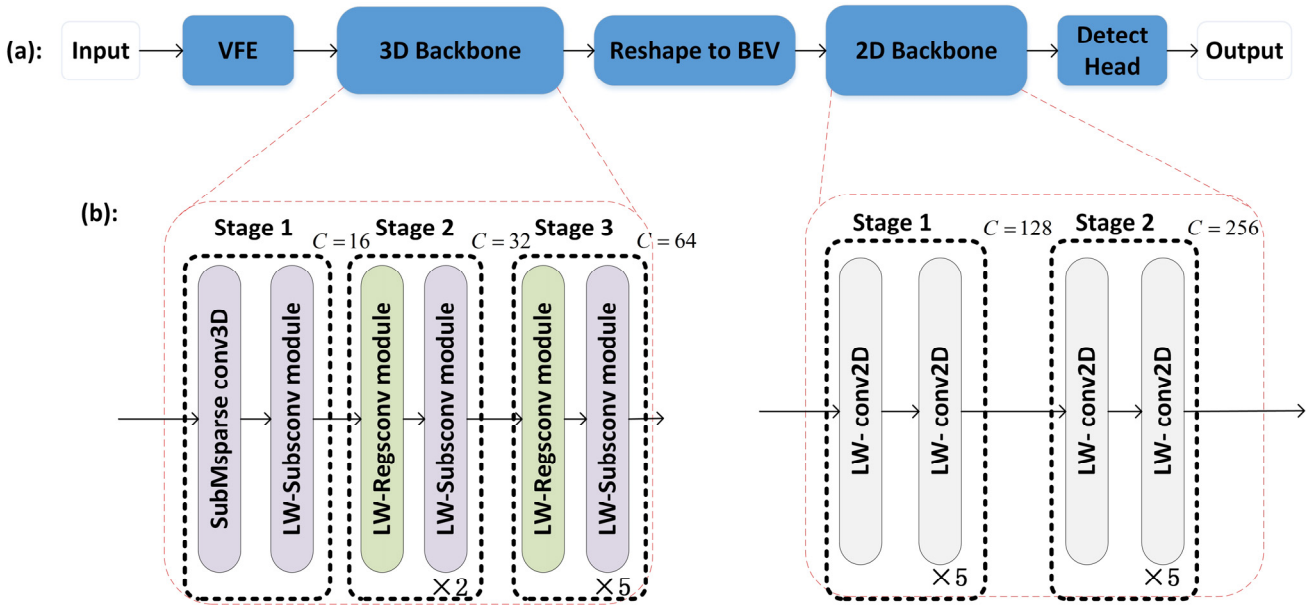


Figure 2. The overall framework of the algorithm. (**a**) Shows the workflow of the algorithm; (**b**) shows the architecture of the 3D backbone and the 2D backbone. C represents the channel of the feature map. $\times n$ represents the number of repeats for the lightweight layer module.

3.3. Loss Function

The loss function for supervised training of the model is the knowledge distillation loss, because knowledge distillation is one of the techniques commonly used to improve the lightweight model. A teacher–student mode is used, where SECOND is used as the teacher model and the lightweight 3D point cloud object detection algorithm proposed in this paper is used as the student model. The overall training framework is shown in

Figure 3, which contains three losses. L_{FSP} constrains the student model to simulate the process of feature map extraction in the teacher model. L_{feature} constrains the student model to simulate the feature map before sending it to the detect head in the teacher model. L_{out} constrains the student model to simulate the soft label in the teacher model. The detailed introduction is below.

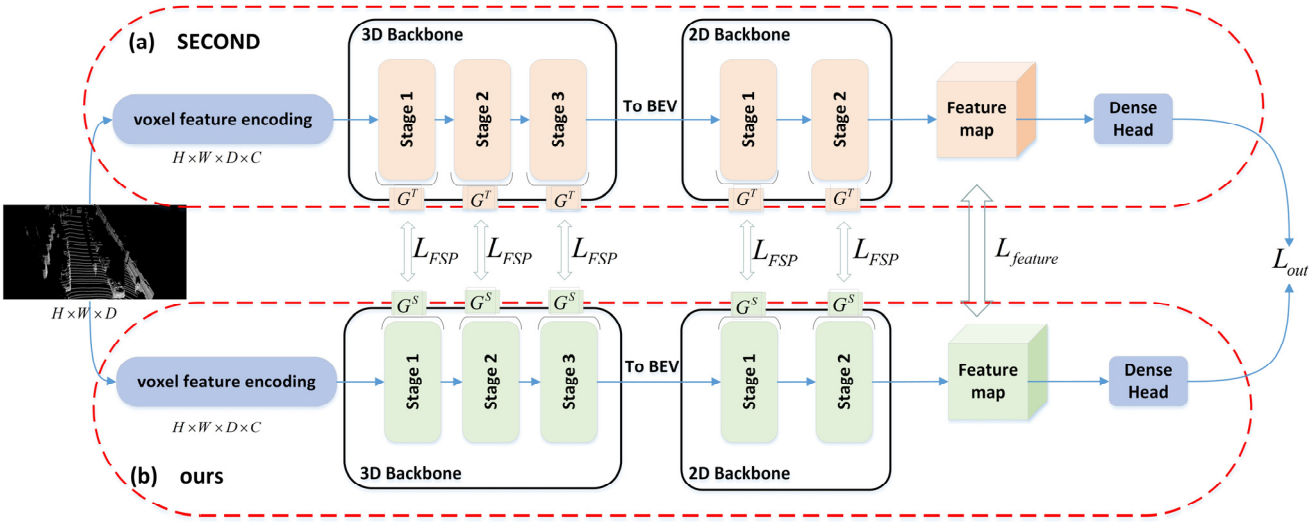


Figure 3. The overall training framework of the algorithm: (**a**) is the teacher model; (**b**) is the student model; G^T represents the FSP matrix of the teacher model; G^S represents the FSP matrix of the student model; $H \times W \times D$ represents the size of the raw point data; $H \times W \times D \times C$ represents the size of the voxel feature.

The calculation of the L_{FSP} needs to define the FSP matrix first. The FSP matrix is used to define the flow of feature information in the backbone. The FSP matrix is used to calculate the dot product of the feature map that is the input and the output in each stage of the backbone. According to [39], the FSP matrix for each stage of the backbone is calculated as follows:

$$G_{i,j} = \sum_{p=1}^{h} \sum_{q=1}^{w} \sum_{r=1}^{d} \left(\frac{F^1_{p,q,r,i} \times F^2_{p,q,r,j}}{h \times w \times d} \right), \tag{1}$$

where i is the number of the channel; $F^1_{p,q,r,i}$ is the value under the coordinate of (h, w, d, i); $F^2_{p,q,r,j}$ is the value under the coordinate of (h, w, d, i); F^1 is the input feature map of the sub-module; F^2 is the output feature map of the sub-module; (h, w, d) is the size of the feature map. L_{FSP} is to calculate the L_2 [40] loss of the student model FSP matrix and the teacher model FSP matrix. The calculation is as follows:

$$L_{FSP} = \frac{1}{2T} \sum_{x} \sum_{i=1}^{n} \lambda_i \times \left\| G_i^T - G_i^S \right\|_2^2, \tag{2}$$

where λ_i represents the weight for each loss term; T is the number of the data; x is the input data; n is the number of the FSP matrix; G_i^T indicates the FSP matrix of x for the teacher model; G_i^S indicates the FSP matrix of x for the student model.

The feature map sent to the detect head can provide rich information for the final prediction. L_{feature} is to calculate the L_2 loss of feature map between the student model and the teacher model. The purpose is to improve the quality of the feature map in the student model and provide better feature information for the final prediction. The calculation is as follows:

$$L_{\text{feature}} = \frac{1}{2T} \sum_{t} \left\| u^T - u^S \right\|_2^2, \tag{3}$$

where t is the index number of the data; T is the number of the data; u^S indicates the feature map of the x^t in the student model; u^T indicates the feature map of the x^t in the teacher model.

L_{out} constrains the student model to learn the soft label of the teacher model, including the boundary box regression of the region proposal network and the classification label of the region classification network. The calculation is as follows:

$$L_{out} = \frac{1}{2T}\sum_{t}\left\|g(x^t) - z(x^t)\right\|_2^2, \tag{4}$$

where t is the index number of the data; T is the number of the data; x^t indicates the t^{th} input data; $g(x^t)$ is the soft label of the x^t in the student model; $z(x^t)$ is the soft label of the x^t in the teacher model.

4. Experiment

4.1. Dataset

This paper conducts experimental verification on two public datasets. The KITTI dataset [41] was co-founded by the Karlsruhe Institute of Technology and the Toyota American Institute of Technology. The dataset contains 7481 training samples and 7518 test samples. The dataset uses the average precision (mAP) to evaluate the object detection model. The nuScenes dataset [42] is a large-scale autonomous driving dataset developed by the Motional team for 3D object detection in urban scenes. The dataset contains 1000 driving scenarios and 390,000 LiDAR-scanned images. The evaluation indexes of the dataset include mAP, NDS, mATE, mASE, mAOE, mAVE, and mAAE. The main evaluation indexes of the dataset are mean average precision (mAP) and nuScenes detection score (NDS). Among them, mAP is calculated by different types of average precision (AP). The average precision (AP) metric defines the matching by the 2D center distance on the threshold ground plane. NDS is the weighted average of mAP and other attribute metrics, including translation, scale, direction, speed, and other box attributes.

4.2. Implementation Details

Taking SECOND [32] as the baseline, this approach is one of the most representative in voxel-based object detection. The architecture of SECOND is end-to-end with good real-time performance. For the training phase, we use an AdamW [43] optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ to optimize the model. The initial learning rate is 0.003 and the weight decay is 0.0001. We train the network with a batch size of four on two NVIDIA Quadro RTX 6000 GPUs. The experiments in this paper are set according to the above; other configurations are the same as the OpenPCDet (https://github.com/open-mmlab/OpenPCDet (accessed on 16 March 2020)) since we conduct all experiments with this toolbox. In this paper, training includes three stages. In the first stage, we minimize the loss function L_{FSP} to make the FSP matrix of the student network similar to that of the teacher network. In the second stage, the model is initialized with the weights obtained from the first training stage. We use L_{feature} to train the feature extraction backbone of the student network. In the third stage, the model is initialized with the weights obtained from the second training stage. We use L_{out} to train the detection head. Finally, we fine-tune the entire network. This fine-tuning process uses the same configurations as the training phase.

4.3. Quantitative Evaluation

Experiment A. In order to verify the feasibility of the lightweight model design in this paper, three different lightweight 3D sparse convolution layer modules are designed for ablation experiments; the experiments are performed on the KITTI dataset.

The first design idea uses the $1 \times 1 \times 1$ filter to process the input data, the number of which is S (where $S < C$); it can reduce the dimension of the input data. Then, the output of the upper layer passes through a $1 \times 1 \times 1$ filter and a $3 \times 3 \times 3$ filter, respectively, the

number of which is $C'/2$. At the end, the $H \times W \times D \times C'$ feature map is concatenated from the output of two group convolutions.

The second design idea uses $3 \times 3 \times 3$ depth-wise sparse 3D convolution to convolve the $H \times W \times D \times C$ input channel by channel, the number of which is C, and then uses $1 \times 1 \times 1$ point-wise sparse 3D convolution to weight the combination of the output of the previous layer in the depth direction, the number of which is S.

The third detailed design idea is described in Section 3.1. The experimental results are shown in Table 1. Comparing SECOND with LW-SECOND-1, it can be seen that the calculation amount is reduced by 17.4 G, the number of parameters is reduced by 1.72 M, and the mAP is reduced by 8%. By comparing SECOND and LW-SECOND-2, it can be seen that the calculation amount is reduced by 49.6 G, the parameter amount is reduced by 4.41 M, and the mAP is reduced by 12.1%. Comparing SECOND and LW-SECOND-3, it can be seen that the calculation amount is reduced by 51.0 G, the parameter amount is reduced by 4.67 M, and the mAP is reduced by 16.4%.

Table 1. Experimental result for the lightweight model design. LW-SECOND-1 indicates the lightweight model designed by the first idea. LW-SECOND-2 indicates the lightweight model designed by the second idea. LW-SECOND-3 indicates the lightweight model designed by the third idea. F and P indicate the number of float operations (/G) and parameters (/M). mAP indicates the accuracy of the detection algorithm.

Dataset	Model	F(G)	P(M)	mAP
KITTI	SECOND	69.8	5.34	63.3
	LW-SECOND-1	52.4	3.62	55.3
	LW-SECOND-2	20.2	0.93	51.2
	LW-SECOND-3	18.5	0.67	48.7

Table 1 shows the FLOPs, parameters, and mAP for the object detection algorithm composed of three lightweight 3D sparse convolution layer modules designed in this paper. The three design methods have different degrees of reduction in the FLOPs and parameters, indicating that the lightweight 3D sparse convolution layer module explored in this paper is feasible and that the third idea is more compressed and better. It shows that the use of group convolution and factorized convolution can effectively compress the model. However, they are accompanied by a certain degree of accuracy reduction.

Experiment B. Knowledge distillation loss is used to supervise the training of the lightweight 3D point cloud object detection algorithm. In order to prove that using the three-part knowledge distillation loss has the best effect on the performance improvement of the algorithm, the ablation experiments are performed on KITTI and nuScences.

Regarding the experiments on KITTI, we report the average precision calculated at 40 sampling recall locations for BEV object detection and 3D object detection, respectively. The implementation details of the experiment are the same as those in Section 4.2. The experimental results are shown in Table 2. The table shows the results of BEV detection and 3D detection in the KITTI dataset. Taking the test results for BEV detection as an example, we compare the SECOND and LW-SECOND-3. The accuracy is reduced from 63.3% to 48.7%. It can be seen that model compression with the lightweight model design reduces the detection accuracy of the algorithm. Compared with LW-SECOND-3 and LW-SECOND-3′, the detection accuracy improves from 48.7% to 48.8%. It can be seen that there is a certain effect, but it is not obvious, indicating that it is not enough to use only the L_{out} constrained student model to simulate the teacher model. Comparing the detection results of LW-SECOND-3′ and LW-SECOND-3′′, it can be seen that the detection accuracy of L_{feature} improves from 48.8% to 53.0% after constraint training, indicating that the feature map of the simulated teacher model has a better effect on the prediction output. Compared with LW-SECOND-3′′ and LW-SECOND-3′′′, the detection accuracy improves from 53.0% to 62.7%. It can be seen that there is a significant improvement in the effect, indicating

that guidance and supervision with the loss of three parts is the best recovery effect on the accuracy of the student model.

Table 2. Experimental results for the KITTI dataset. Model indicates which network architecture is selected; LW-S-3 represents LW-SECOND-3, indicating that no knowledge distillation is used. The symbol $'$ indicates that only L_{out} is used for knowledge distillation; the symbol $''$ indicates that L_{out} and L_{feature} are used for knowledge distillation; the symbol $'''$ indicates that L_{out}, L_{feature}, and L_{FSP} are used for knowledge distillation.

Task	Model	Car			Pedestrians			Cyclists			mAP
		Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mod	Hard	
BEV	SECOND	88.07	79.37	77.95	55.10	46.27	44.76	73.67	56.04	48.78	63.3
	LW-S-3	69.81	61.72	57.86	44.28	38.53	33.61	54.49	41.19	36.64	48.7
	LW-S-3$'$	70.62	61.57	58.61	45.09	37.34	31.43	55.27	40.93	37.46	48.8
	LW-S-3$''$	76.49	66.24	63.93	48.16	41.87	33.69	62.51	44.23	39.75	53.0
	LW-S-3$'''$	87.37	79.12	77.81	53.94	45.65	43.56	73.39	56.11	47.63	62.7
3D	SECOND	83.13	73.66	66.20	51.07	42.56	37.29	70.51	53.85	46.90	58.4
	LW-S-3	68.45	60.97	57.51	41.38	33.87	29.06	58.82	44.16	36.21	47.0
	LW-S-3$'$	69.54	61.20	56.07	40.26	33.04	30.54	56.32	41.51	37.62	47.2
	LW-S-3$''$	73.85	64.51	58.28	44.57	36.25	32.86	61.63	45.72	39.93	50.8
	LW-S-3$'''$	82.73	73.39	65.16	51.43	41.93	37.74	69.54	53.60	45.81	57.9

Regarding the experiments on nuScenes, we report the detection results in terms of the evaluation indicators mAP, NDS, mATE, mASE, mAOE, mAVE, and mAAE. The implementation details of the experiment are the same as those in Section 4.2. The experimental results are shown in Table 3. The table shows the evaluation of algorithm performance by different evaluation indicators. The larger the values of mAP and NDS, the better; the smaller the values of mATE, mASE, mAOE, mAVE, and mAAE, the better. Compared with LW-SECOND-3 and LW-SECOND-3$'''$, we can see that mAP and NDS increase by 10.3% and 9.4%, respectively.

Table 3. Experimental results for the nuScences dataset. Model indicates which network architecture is selected; KD indicates whether knowledge distillation is used; L_{out}, L_{feature}, and L_{FSP} indicate which knowledge transfer is used.

Model	KD	L_{out}	$L_{feature}$	L_{FSP}	mAP	NDS	mATE	mASE	mAOE	mAOVE	mAAE
SECOND	-	-	-	-	52.8	63.3	-	-	-	-	-
LW-SECOND-3	-	-	-	-	40.5	51.4	45.2	28.8	52.6	34.9	19.6
LW-SECOND-3$'$	✓	✓	×	×	41.3	52.6	44.9	28.6	51.3	32.7	18.4
LW-SECOND-3$''$	✓	✓	✓	×	43.5	56.4	39.1	26.5	43.2	30.8	20.6
LW-SECOND-3$'''$	✓	✓	✓	✓	50.8	60.8	31.9	25.4	32.1	26.4	19.9

Figure 4 also provides the detailed prevision–recall (PR) curves for two datasets, showing the difference between the model after the lightweight model design with knowledge distillation and without knowledge distillation. It can be seen in the two ablation experiments that the loss function in this paper can effectively restore the accuracy of the lightweight model by using the three-part knowledge distillation losses to supervise and train the lightweight 3D point cloud object algorithm. It ensures that the lightweight model can meet the task requirements while reducing the complexity of the model.

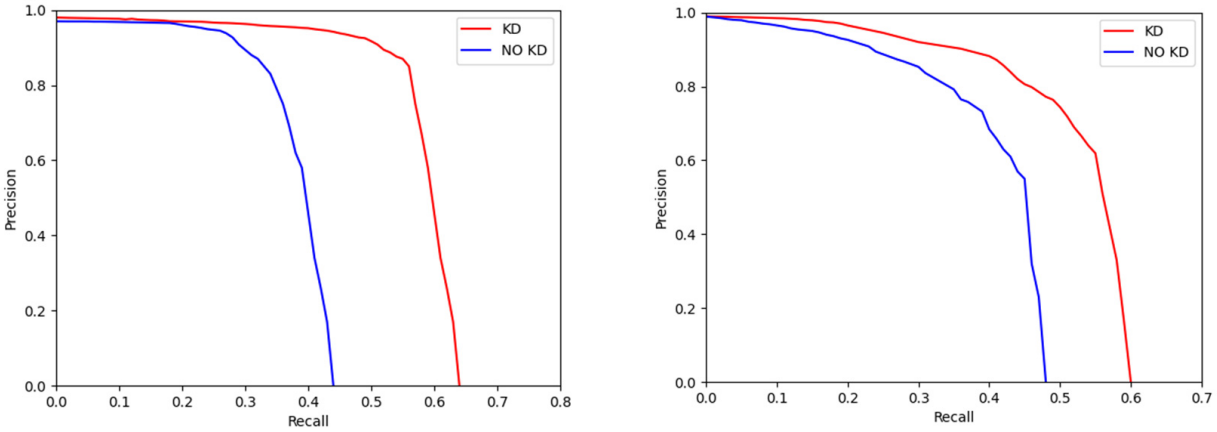


Figure 4. PR curves of two datasets. (**a**) The results of the KITTI dataset; (**b**) the results of the nuSences dataset.

Experiment C. In order to prove the necessity of model compression, the 3D object detection algorithm proposed in this paper is compared with other 3D object detection algorithms. The comparison results are shown in Table 4. It can be seen that, compared with other algorithms, the parameters and FLOPs of the algorithm proposed in this paper are extremely reduced and that the detection accuracy is maintained at the upper and middle levels. Obviously, the method proposed in this paper is more suitable for deployment on edge devices. Table 4 shows the results with AP calculated by recall 40 positions for car class on the KITTI test set. F and P indicate the number of float operations (/G) and parameters (/M).

Table 4. Experimental results for 3D point cloud object detection.

Dataset	Model	Car			F(G)	P(M)
KITTI	PointPilliar [44]	82.58	74.31	68.99	34.3	4.8
	SECOND [32]	83.13	73.66	66.20	69.8	5.3
	PointRCNN [30]	84.32	75.42	67.86	104.9	4.1
	Ours	82.73	73.39	65.16	18.8	0.7

4.4. Qualitative Results

In this section, we visualize the results of 3D object detection on KITTI and nuScences. It shows the influence of whether to use the knowledge distillation loss for supervised training on the lightweight 3D point cloud object detection algorithm. Figure 5 shows some detection results from the KITTI test set. Figure 6 shows some detection results from the nuScences test set. Through comparison, it can be seen that the lightweight 3D point cloud object detection algorithm using knowledge distillation is more accurate. The purple circle shows the area where the error classification is reduced. The reason is mainly due to the L_{feature} lightweight 3D point cloud object detection algorithm, which simulates the feature map before feeding it into the detect head in SECOND. The learning of the feature map provides more information for the output prediction and improves the classification accuracy. The red circle shows the area where the false positive prediction is reduced. The main reason is that L_{FSP}, the lightweight 3D point cloud object detection algorithm, simulates the process of feature extraction in SECOND and knows how to extract more effective features to provide useful feature maps for subsequent detection.

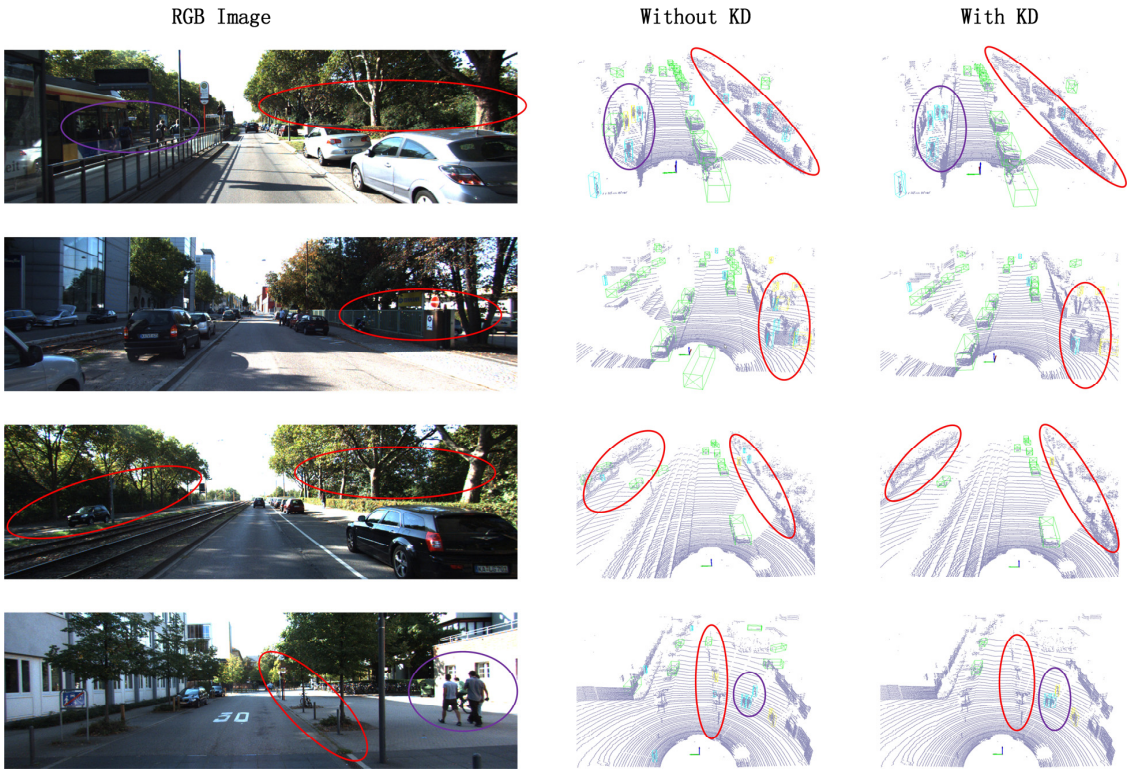


Figure 5. Qualitative results on the KITTI dataset. The figure shows the RGB image of the autonomous driving scene and the detection results of LW-SECOND-3 trained without and with knowledge distillation. The green box represents the car, the blue box represents the pedestrian, and the yellow box represents the cyclist. The circles mark the detection areas that are effectively enhanced using knowledge distillation.

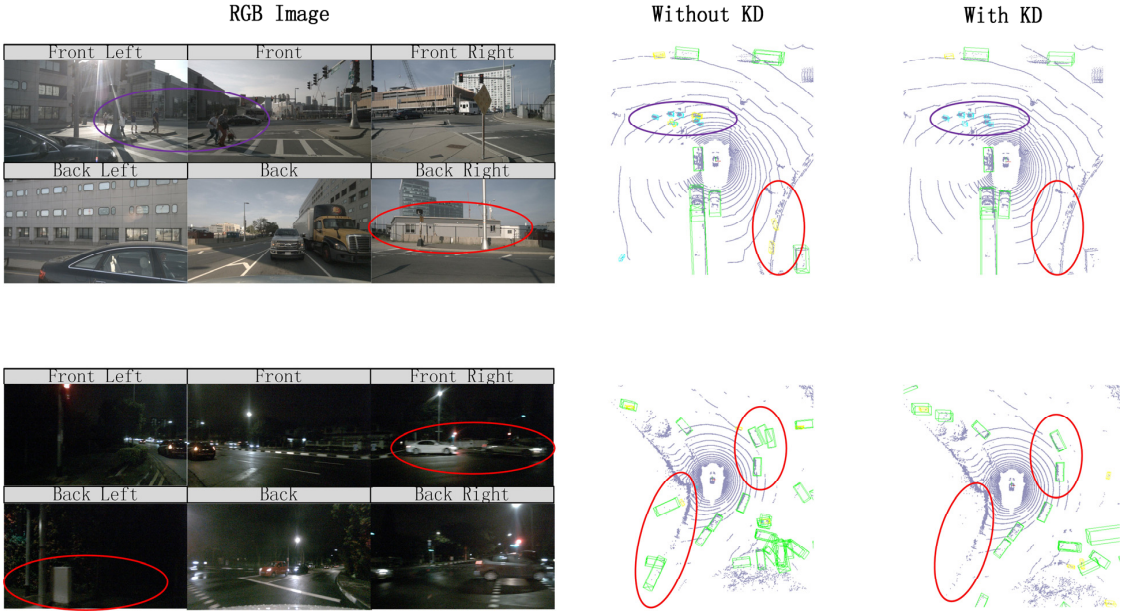


Figure 6. Qualitative results on the nuScences dataset. The figure shows the RGB image of the autonomous driving scene and the detection results of LW-SECOND-3 trained without and with knowledge distillation. The green box represents the car, the blue box represents the pedestrian, and the yellow box represents the cyclist. The circles mark the detection areas that are effectively enhanced using knowledge distillation.

5. Conclusions

In order to reduce the complexity of the model and make it meet the needs of our edge devices, this paper proposes a lightweight 3D point cloud object detection algorithm. Firstly, a novel 3D sparse convolution layer module is designed by using factorized convolution and group convolution, which is used as the construction block of a lightweight convolution network. The module is composed of point-wise 3D convolution and depth-wise 3D sparse convolution. At the same time, transpose and reshape operations are introduced to process the feature map and help information flow between channels. It reduces the complexity of the model and accelerates it. In addition, this paper is inspired by the knowledge distillation, using the teacher–student mode to complete training. Extensive experimental verification is carried out on two public datasets. The effectiveness of improving the detection accuracy of lightweight models is proven through quantitative evaluation and qualitative results. Finally, the algorithm proposed in this paper is compared with the baseline, which shows that the complexity of the model is greatly reduced while the detection accuracy is not significantly reduced. Compared with other 3D point cloud object detection algorithms, the 3D point cloud object detection algorithm in this paper achieves a better balance between complexity and detection accuracy. It indicates that the algorithm proposed in this paper is more suitable for deployment on edge devices. The lightweight 3D point cloud object detection algorithm explored in this paper has certain theoretical and practical significance. In the future, it can be extended to other 3D tasks to reduce model complexity and achieve model acceleration.

Author Contributions: Conceptualization, Z.L. (Ziyi Li) and Y.L.; methodology, Z.L. (Ziyi Li) and Y.L.; software, G.X.; validation, Z.L. (Ziyi Li) and Y.L.; formal analysis, Y.W.; investigation, H.Q.; resources, H.Q., Y.W. and Y.L.; data curation, G.X.; writing—original draft preparation, Z.L. (Ziyi Li); writing—review and editing, Z.L. (Ziyi Li), Z.L. (Zhuoyang Lyu) and Y.L.; visualization, Y.W.; supervision, H.Q.; project administration, Y.W. and Y.L.; funding acquisition, Y.W. and Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China under grant 62131001 and 61971456, by Beijing Municipal Natural Science Foundation under grant 4232003, and by Yuyou Talent Training Program under grant 218051360020XN115/014 of the North China University of Technology.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We thank the anonymous reviewers for their good suggestions and comments to help improve the quality of the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X.; Li, H. PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 10526–10535. [CrossRef]
2. Han, S.; Mao, H.; Dally, W.J. Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding. In Proceedings of the International Conference on Learning Representions, San Juan, Puerto Rico, 2–4 May 2016.
3. Liu, Z.; Mu, H.; Zhang, X.; Guo, Z.; Yang, X.; Cheng, T.K.-T.; Sun, J. MetaPruning: Meta Learning for Automatic Neural Network Channel Pruning. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3295–3304.
4. Louizos, C.; Welling, M.; Kingma, D.P. Learning Sparse Neural Networks through L0 Regularization. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
5. Choi, J.; Wang, Z.; Venkataramani, S.; Chuang, P.I.-J.; Srinivasan, V.; Gopalakrishnan, K. PACT: Parameterized Clipping Activation for Quantized Neural Networks. *arXiv* **2018**, arXiv:1805.06085.

6. Dong, R.; Tan, Z.; Wu, M.; Zhang, L.; Ma, K. Finding the task-optimal low-bit sub-distribution in deep neural networks. In Proceedings of the International Conference on Machine Learning, Baltimore, MD, USA, 17–23 July 2022.
7. Nagel, M.; van Baalen, M.; Blankevoort, T.; Welling, M. Finding the task-optimal low-bit sub-distribution in deep neural networks. In Proceedings of the International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019.
8. Howard, A.; Sandler, M.; Chu, G.; Chen, L.-C.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V.; et al. Searching for MobileNetV3. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324.
9. Ma, N.; Zhang, X.; Zheng, H.-T.; Sun, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 116–131.
10. Chollet, F. Xception: Deep learning depth-wise separable convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1251–1258.
11. Hinton, G.E.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *arXiv* **2015**, arXiv:1503.02531.
12. Zagoruyko, S.; Komodakis, N. Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer. *arXiv* **2017**, arXiv:1612.03928.
13. Tian, Y.; Krishnan, D.; Isola, P. Contrastive representation distillation. *arXiv* **2019**, arXiv:1910.10699.
14. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9. [CrossRef]
15. 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, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6848–6856.
16. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. *arXiv* **2016**, arXiv:1602.07360.
17. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
18. Jin, J.; Dundar, A.; Culuricello, E. Flattened convolutional neural networks for feedforward acceleration. *arXiv* **2014**, arXiv:1412.5774.
19. Xie, S.; Girshick, R.; Dpllár, P.; Tuand, Z.; He, K. Aggregated residual transformations for deep neural network. *arXiv* **2016**, arXiv:1611.05431.
20. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
21. Tang, P.; Wang, H.; Kwong, S. G-MS2F: GoogLeNet based multi-stage feature fusion of deep CNN for scene recognition. *Neurocomputing* **2017**, *225*, 188–197. [CrossRef]
22. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
23. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]
24. Cao, Q.; Shen, L.; Xie, W.; Parkhi, O.M.; Zisserman, A. Vggface2: A dataset for recognising faces across pose and age. In Proceedings of the 13th IEEE International Conference on Automatic Face & Gesture Recognition, Xi'an, China, 15–19 May 2018; pp. 67–74.
25. Bhandari, M.; Yogarajah, P.; Kavitha, M.S.; Condell, J. Exploring the Capabilities of a Lightweight CNN Model in Accurately Identifying Renal Abnormalities: Cysts, Stones, and Tumors, Using LIME and SHAP. *Appl. Sci.* **2023**, *13*, 3125. [CrossRef]
26. Li, W.; Zhang, L.; Wu, C.; Cui, Z.; Niu, C. A new lightweight deep neural network for surface scratch detection. *Int. J. Adv. Manuf. Technol.* **2022**, *123*, 1999–2015. [CrossRef] [PubMed]
27. Charles, R.Q.; Su, H.; Kaichun, M.; Guibas, L.J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 77–85. [CrossRef]
28. Zhou, Y.; Tuzel, O. VoxelNet: End-to-End learning for Point Cloud Based 3D Object Detection. *arXiv* **2017**, arXiv:1711.06396.
29. Lang, A.H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; Beijbom, O. PointPillars: Fast Encoders for Object Detection From Point Clouds. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 12689–12697. [CrossRef]
30. Shi, S.; Wang, X.; Li, H. PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 16–20.
31. Zhang, Y.; Zhang, Q.; Zhu, Z.; Hou, J.; Yuan, Y. GLENet: Boosting 3D Object Detectors with Generative Label Uncertainty Estimation. *arXiv* **2022**, arXiv:2207.02466.
32. Yan, Y.; Mao, Y.; Li, B. Second: Sparsely embedded convolutional detection. *Sensors* **2018**, *18*, 3337. [CrossRef] [PubMed]
33. Zhang, L.; Dong, R.; Tai, H.; Ma, K. PointDistiller: Structured Knowledge Distillation Towards Efficient and Compact 3D Detection. *arXiv* **2022**, arXiv:2205.11098.

34. Li, Y.; Chen, H.; Cui, Z.; Timofte, R.; Pollefeys, M.; Chirikjian, G.S.; Van Gool, L. Towards efficient graph convolutional networks for point cloud handing. In Proceedings of the International Coference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 3752–3762.
35. Liu, Z.; Tang, H.; Lin, Y.; Han, S. Point-voxel cnn for efficient 3d deep learning. In Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2016.
36. Tang, H.; Liu, Z.; Zhao, S.; Lin, Y.; Lin, J.; Wang, H.; Han, S. Searching efficient 3d architectures with sparse point-voxel convolution. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; Volume 3, pp. 685–702.
37. Gwak, J.Y.; Choy, C.; Savarese, S. Generative sparse detection networks for 3D single-shot object detection. In Proceedings of the IEEE/CVF European Conference on Computer Vision(ECCV), Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 297–313.
38. Danila, R.; Anna, V.; Anton, K. Fcaf3D: Fully convolutional anchor-free3d object detection. In Proceedings of the IEEE/CVFEuropeanConference on Computer Vision(ECCV), Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 477–493.
39. Yim, J.; Joo, D.; Bae, J.; Kim, J. A gift from knowledge distillation: Fast optimization, network minimization and transfer learn-ing. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4133–4141.
40. Li, Q.; Jin, S.; Yan, J. Mimicking Very Efficient Network for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 7341–7349. [CrossRef]
41. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]
42. 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 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 11618–11628.
43. Loshchilov, I.; Hutter, F. Decoupled weight decay regularization. In Proceedings of the International Conference on Learning Representations(ICLR), New Orleans, LA, USA, 6–9 May 2019.
44. Paigwar, A.; Sierra-Gonzalez, D.; Erkent, Ö.; Laugier, C. Frustum-pointpillars: A multi-stage approach for 3d object detection using rgb camera and lidar. In Proceedings of the International Conference of Computer Vision, ICCV'2021, Montreal, BC, Canada, 11–17 October 2021; Volume 3, pp. 2926–2933.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.