RESEARCH ARTICLE

Improved Object Detection Algorithm Based on Multi-scale and Variability Convolutional Neural Networks

Jiaxun Yang[1,*] **and Yilihamujiang Gapar**[2]

[1] Lyceum of the Philippines University, Batangas 4200, Philippines
[2] Department of Student Affairs, Northwest Minzu University, Lanzhou 730030, China

Abstract

This paper proposes an improved object detection algorithm based on a dynamically deformable convolutional network (D-DCN), aiming to solve the multi-scale and variability challenges in object detection tasks. First, we review traditional methods in the field of object detection and introduce the current research status of improved methods based on multi-scale and variability convolutional neural networks. Then, we introduce in detail our proposed improved algorithms, including an improved feature pyramid network and a dynamically deformable network. In the improved feature pyramid network, we introduce a multi-scale feature fusion mechanism to better capture target information at different scales. In dynamically deformable networks, we propose dynamic offset calculations and dynamic convolution operations to achieve dynamic adaptation to the target shape and pose. We also validate our method by conducting experiments on the datasets KITTI and Caltech. Finally, we design a comprehensive loss function that considers both location localization error and category classification error to guide model training. Experimental results show that our improved algorithm achieves significant performance improvements in target detection tasks, with higher accuracy and robustness compared with traditional methods. Our work provides an effective method to solve the multi-scale and variability challenges in target detection tasks and has high practical value and prospects for general application.

Keywords: object detection, feature pyramid network, multi-scale fusion, dynamic convolution, KITTI and Caltech.

1 Introduction

In the field of computer vision, object detection is a key task that involves identifying and locating single or multiple objects in an image. This technology has a wide range of application scenarios, including but not limited to autonomous driving, video surveillance, human-computer interaction, and industrial automation. Especially in the fields of autonomous driving and safety monitoring, object detection plays a crucial role. For example, in autonomous driving systems, accurate target detection

Academic Editor:
Teerath Kumar

Submitted: 01 December 2023
Accepted: 16 May 2024
Published: 21 May 2024

Vol. 1, **No.** 1, 2024.
10.62762/TETAI.2024.115892

***Corresponding author:**
Jiaxun Yang
yangjxjason@gmail.com

Citation
Yang, J., & Gapar, Y. (2024). Improved Object Detection Algorithm Based on Multi-scale and Variability Convolutional Neural Networks. *IECE Transactions on Emerging Topics in Artificial Intelligence*, 1(1), 31–43.

can not only identify vehicles, pedestrians, and obstacles on the road but also interpret traffic signs and signals to ensure driving safety. In the field of video surveillance, target detection technology can help the surveillance system effectively identify and track specific targets, which is crucial for improving public safety and preventing criminal activities [37][23].

Before deep learning became popular, object detection mainly relied on hand-designed features and traditional machine learning techniques. A common method is to use sliding window technology with feature descriptors such as SIFT (Scale Invariant Feature Transform) [22], SURF (Speed-up Robust Features) [1] or HOG (Histogram of Oriented Gradients) [25]. Sliding window technology moves windows of different sizes on the image, extracts features on each window, and then uses a classifier such as SVM (Support Vector Machine) [14] to identify and classify targets.

For example, the HOG feature proposed by Dalal and Triggs in 2005 combined with linear SVM was one of the early benchmark methods for pedestrian detection. The HOG feature captures the shape information of the target by calculating the gradient direction histogram of the local area of the image, which is effective for single-scale pedestrian detection. However, when the target size is variable or the background is complex, the effect of this method will drop significantly, because hand-designed features often lack adaptability to complex changes. Although object detection technology has made significant progress, it still faces several challenges. Traditional target detection methods work well in simple scenes but often perform poorly in complex environments.

In the field of target detection, existing deep learning methods for dealing with multi-scale and variability problems mainly focus on improving the network architecture to adapt to targets of different scales. Feature Pyramid Network (FPN) [20] is an effective solution. It achieves the fusion of high-level semantic information and low-level detailed information by establishing a top-down information flow and generating feature maps at different levels. In this way, the feature maps at each level can correspond to targets of different sizes, greatly improving the detection ability of small targets. In addition, the deformable convolutional network (DCN) [4] introduces learnable offsets to enable the convolution kernel to adapt to the specific shape and posture of the target, enhancing the network's adaptability to complex shape and posture changes. The application of these technologies significantly improves the model's performance in multi-scale and high-variability environments, while also maintaining a high recognition rate for large-sized targets. These advances not only solve the problems encountered by traditional methods in complex scenes but also promote the development of target detection technology in a more efficient and accurate direction.

However, these existing deep learning methods still have shortcomings in handling multi-scale and variability objects. On the one hand, traditional convolutional networks are usually sensitive to the size of the input image, and their performance is often affected when encountering targets with large size changes. On the other hand, complex background and environmental factors (such as lighting changes, occlusions, and background clutter) can also reduce detection accuracy. Therefore, developing a target detection algorithm that can adapt to multi-scale changes and have high environmental adaptability has become a research hotspot.

This paper proposes an improved target detection algorithm based on multi-scale and variability convolutional neural networks. The algorithm is specially designed with a multi-scale feature fusion mechanism that can effectively integrate information from different convolutional layers, thereby improving the detection capabilities of small and large-sized targets. At the same time, we introduced a variable convolution structure, which dynamically adjusts the convolution kernel parameters to cope with different detection scenarios and enhances the model's adaptability to complex environments.

To verify the effectiveness of the proposed algorithm, we conduct extensive experiments on three public datasets, KITTI [8], NEXET and Caltech [6]. The KITTI dataset is widely used for visual tasks related to autonomous driving, and contains vehicle and pedestrian detection data in a variety of traffic scenarios; while the Caltech dataset focuses on pedestrian detection, covering various pedestrian sizes and complex background conditions. Through testing on these two data sets, we not only demonstrated the efficient performance of the algorithm in general target detection tasks but also specifically verified its superiority in multi-scale and high-variation environments.

Our contributions mainly include:

1 Developed a new multi-scale feature fusion strategy, which significantly improves the detection accuracy of targets of various sizes by precisely controlling the integration of information flow between different scales.

2 Designed a variable convolution module that uses a learnable parameter adjustment strategy to enable the network to automatically adjust its feature extraction method according to the characteristics of the target and environmental conditions.

3 Comprehensive experiments are conducted on KITTI and Caltech datasets, and the results demonstrate that our model has significant performance advantages over existing technologies when handling target detection tasks in multi-scale and complex environments.

Through these innovations, our algorithm not only performs well on standard data sets but also demonstrates its wide applicability and practical value in practical application scenarios, providing new technical solutions for target detection in complex environments.

2 Related Work

In the field of target detection, with the continuous advancement of technology, researchers have developed a variety of complex methods to deal with various challenges, especially for the detection of multi-scale and deformed targets. These methods are not only innovative in theory, but also show extremely high efficiency and accuracy in practical applications. Below is a detailed extended introduction to these methods, divided into three main categories: region-based methods, regression-based methods, and structural optimization/enhancement methods.

2.1 Region-based Approach

R-CNN [11] extracts deep feature representations by applying high-capacity convolutional neural networks to each independent candidate region. Although this method is effective, it is computationally expensive and slow. Fast R-CNN [10] improves this process and introduces a RoI pooling layer, which can quickly extract the features of each region from a unified feature map of the entire image, significantly improving the speed and efficiency while passing Using softmax instead of SVM simplifies the training process. Faster R-CNN [30] further optimizes the generation process of candidate regions by integrating a Region Proposal Network (RPN), which can be jointly trained with the detection network end-to-end, further improving speed and accuracy. Mask R-CNN [13]: Based on Faster R-CNN, a branch is added to generate a pixel-level mask of the target, which enables the model to not only perform target detection but also perform more refined instance segmentation. This method is particularly suitable for application scenarios that require precise target contour information, such as medical image analysis and video editing.

2.2 Regression-based Approach

The regression-based method handles target detection in a more intuitive and fast way, directly predicting the location and category of the target on the entire image, greatly improving the processing speed.

YOLO [29] series: Especially YOLOv1 to YOLOv4, each generation has made significant improvements based on the previous generation, such as deeper networks, better feature utilization strategies, and more efficient framework design. YOLO transforms the target detection problem into a single regression problem, achieving very fast detection speed, and is very suitable for occasions that require real-time processing. SSD [24]: This algorithm effectively handles the multi-scale target detection problem by predicting the location and category of the target on multiple feature maps of different scales. The design of the SSD allows it to achieve very good speed performance while maintaining high accuracy. EfficientDet [32]: This algorithm further balances the depth, width, and input image resolution of the network through compound scaling technology, improving efficiency and accuracy. EfficientDet shows excellent performance on multiple standard datasets, especially on resource-constrained devices.

2.3 Structural Enhancement Approach

In order to adapt to changes in multi-scale and target morphology, some methods improve the adaptability and accuracy of the model through structural optimization.

Feature Pyramid Network (FPN): By establishing multi-level feature maps, each layer can extract information at different scales. FPN can effectively enhance the model's detection capabilities for targets of various sizes. This top-down structure provides rich semantic information and detailed information for target detection. Deformable Convolutional Networks (DCN): By allowing the shape of the convolution kernel to dynamically adapt to the characteristics of the input data, DCN provides the possibility to handle irregularly shaped targets, which is difficult to achieve in traditional convolutional networks. RetinaNet [21]: By introducing Focal Loss, it solves the imbalance problem of positive and negative samples often encountered in target detection, especially when there are a large number of easy-to-classify background samples, and improves the detection performance of small targets.

Each of the above methods has its own merits, but what they have in common is that they have greatly promoted the development of target detection technology, improved the accuracy, speed, and adaptability of detection, and met the needs of different platforms from mobile devices to high-performance servers.

3 Methodology

To effectively address the challenges of multi-scale and high variability, our research work introduces two innovative techniques: improved feature pyramid network (iFPN) and dynamically deformable convolutional network (D-DCN). These methods specifically solve the key issues of size adaptation and shape variation in target detection and significantly improve the performance and adaptability of the detection algorithm through precise mathematical formulas and complex network design. Our overall network architecture is shown in Figure 1 and Algorithm 1 shows our process.

3.1 Improved Feature Pyramid Network

In order to further improve the effect of multi-scale target detection, we developed the improved feature pyramid network(IFPN), which is a network architecture specially designed to improve the efficiency and quality of information fusion between feature layers. IFPN not only comprehensively utilizes features from different layers, but also introduces a novel adaptive weight adjustment mechanism, allowing the network to automatically adjust its contribution to the final detection task based on the importance of each layer's features.

Semantic Score Assessment. The core of IFPN lies in the semantic evaluation of each feature layer, which determines the weight of this layer in feature fusion. The semantic evaluation is based on the assumption that deeper networks usually capture higher-level semantic information, which is especially important for recognizing small-sized objects. We analyze the output

```
Algorithm 1: D-DCN
Input: Feature Map $x$, Weights $W_p$, $W_d$,
       Parameters $\theta$, $\phi$, $\gamma$, $\lambda$
Output: Detection Results
Initialize weights $W$ and parameters $\theta$, $\phi$, $\gamma$, $\lambda$;
for each feature map $F_i$ in $x$ do
    Compute semantic score:
      $S(F_i) = \sigma(\text{Conv}(F_i, W_s) + b_s)$;
    Compute adaptive weight: $w_i = \frac{\exp(\gamma \cdot S(F_i))}{\sum_j \exp(\gamma \cdot S(F_j))}$;
end
Compute merged feature:
  $F_{\text{merged}} = \sum_i w_i \times U(F_i, \text{size})$;
for each position $p$ in $F_{merged}$ do
    Compute position offset:
      $\Delta p = \theta \times \tanh(\text{Conv}(x, W_p))$;
    Compute shape offset:
      $\Delta s = \phi \times \sigma(\text{Conv}(x, W_d))$;
    for each kernel position $p_n$ in $R$ do
        Adjusted position: $p_{\text{adj}} = p + p_n + \Delta p_n$;
        Adjusted kernel: $W_{\text{adj}} = \text{Conv}(p_n, \theta, \phi)$;
        Convolution output:
          $y(p) += x(p_{\text{adj}}) \times W_{\text{adj}}$;
    end
end
return Detection Results
```

of each feature layer through a deep convolutional network and calculate its semantic score. The process can be expressed as:

$$S(F_i) = \sigma(Conv(F_i, W_s) + b_s) \tag{1}$$

where F_i is the feature map Conv of the i-th layer, represents the convolution operation W_s and b_s are the weight and bias of the convolution layer respectively, and σ is usually a nonlinear activation function, such as ReLU [12] or Sigmoid, to ensure the output The score is within a reasonable range.

Dynamic Weight Adjustment. Based on the semantic score of each layer, we calculate dynamic weights to adjust the influence of each layer's features in the fusion process. This dynamic adjustment mechanism allows the network to automatically optimize the feature fusion strategy based on different inputs, thereby better-handling targets of various sizes. This process can be expressed as:

$$w_i = \frac{\exp(\gamma \cdot S(F_i))}{\sum_j \exp(\gamma \cdot S(F_j))} \tag{2}$$

where γ is a trainable scaling parameter used to adjust the sensitivity of the softmax function, thereby

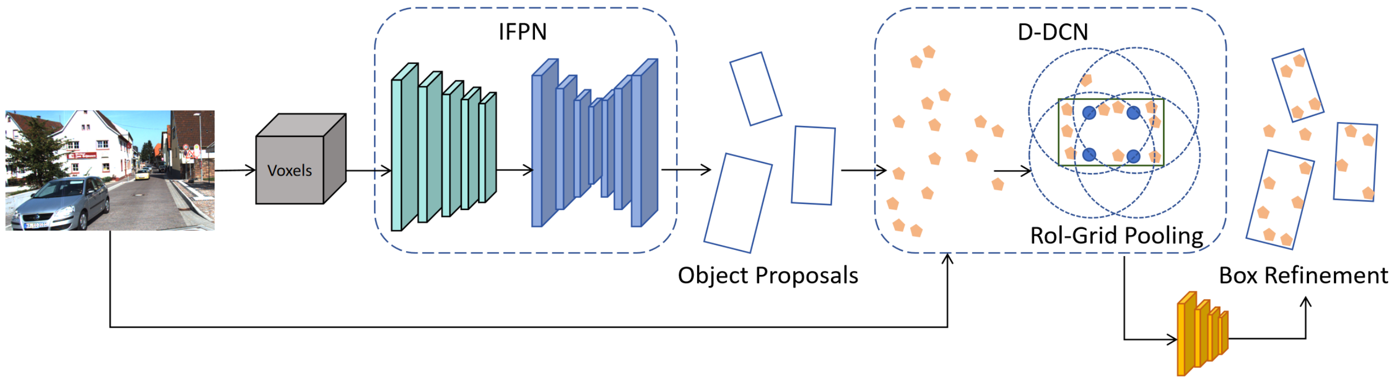


Figure 1. A diagram of our overall network architecture.

affecting the weight distribution of different feature layers.

Feature Fusion Strategy. Using the calculated weights, we perform a weighted fusion of features from all layers to generate the final feature map for object detection. This step is accomplished through upsampling and weighted summation, ensuring that all feature maps contribute to the final detection result. The fusion process is:

$$F_{\text{merged}} = \sum_i w_i \cdot U(F_i, \text{size}) \tag{3}$$

Among them, $U(F_i, \text{size})$ is an upsampling operation, which upsamples each feature map to a uniform size for effective fusion.

Through this improved feature pyramid network design, iFPN can more effectively utilize features from different network depths and optimize the detection capabilities of multi-scale targets. This not only enhances the model's sensitivity to small-sized targets but also improves its ability to adapt to changes in background complexity. In addition, the adaptive characteristics of IFPN enable it to automatically adjust the feature fusion strategy according to specific tasks and data sets, further improving the versatility and flexibility of the model. These properties make IFPN a powerful tool for multi-scale object detection scenarios ranging from simple to extremely complex.

3.2 Dynamically Deformable Convolutional Network

In object detection, a dynamically deformable convolutional network (D-DCN) is a novel technology designed to solve the problem of irregular shape and pose changes of objects. Traditional fixed convolution kernels perform poorly in handling such changes because they cannot adapt to the local structure and shape changes of the target. In order to solve this problem, D-DCN introduces a dynamic deformation mechanism that enables the convolution kernel to adjust dynamically according to the specific shape of the target, thereby more accurately capturing the detailed information of the target.

Dynamic Offset Calculation. In dynamically deformable convolutional networks (D-DCN), the calculation of dynamic offsets is one of the key steps. These offsets allow the convolution kernel to adjust dynamically according to the local structure and shape changes of the target to more accurately capture the detailed information of the target. We will introduce the calculation process of position offset and shape offset respectively.

Position Offset Calculation: The position offset is used to adjust the position of the convolution kernel to adapt to the position change of the target. This process utilizes a convolutional layer and a nonlinear function to generate position offsets. Expressed in the following form:

$$\Delta p = \theta \cdot \tanh(\text{Conv}(x, W_p)) \tag{4}$$

where Δp represents the position offset, θ is the learnable parameter Conv represents the convolution operation, and x is the input feature map W_p is the weight of the convolution layer. Through this formula, the network can learn the position offsets at different positions based on the input feature map x.

Shape Offset Calculation: Shape offset is used to adjust the shape of the convolution kernel to adapt to changes in the shape of the target. This process also uses a convolutional layer and a nonlinear function to generate shape offsets. The process can be expressed as follows:

$$\Delta s = \phi \cdot \sigma(\text{Conv}(x, W_s)) \tag{5}$$

where Δs represents the shape offset, ϕ is the learnable parameter, σ is a nonlinear activation function, Conv represents the convolution operation, x is the input feature map, and W_s is the weight of the convolution layer. Through this formula, the network can learn the shape offsets at different positions based on the input feature map x.

Through the calculated position offset and shape offset, D-DCN can dynamically adjust the position and shape of the convolution kernel during the convolution operation, thereby better capturing the local structure and shape changes of the target.

3.3 Dynamic Convolution Operation

The dynamic convolution operation in the dynamically deformable convolutional network (D-DCN) is one of its core components. It enables the network to dynamically adjust the position and shape of the convolution kernel according to the local structure and shape changes of the target, thereby making it more precise. Capture the characteristics of the target well.

Generation of Dynamic Convolution Kernel. In dynamically deformable convolutional networks (D-DCN), the generation of dynamic convolutional kernels is a key task. In traditional convolution operations, the weight of the convolution kernel is fixed, but in D-DCN, the weight of the convolution kernel is adjusted according to the dynamic offset to adapt to the irregular shape and attitude changes of the target.

Offset-based Dynamic Convolution Kernel. The generation of dynamic convolution kernel is based on the adjustment of offset. During the convolution process, for each convolution kernel position p, it is adjusted according to the learned offset to adapt to the local structure and shape changes of the target. Specifically, the weight $W(p_n, \theta, \phi)$ of the dynamic convolution kernel can be calculated by the following formula:

$$W(p_n, \theta, \phi) = Conv(p_n, \theta, \phi) \tag{6}$$

In this formula, p_n is the position index of the convolution kernel, and θ and ϕ are learnable parameters, which are used to control the dynamic adjustment of the convolution kernel weight. Conv represents a convolution operation. Through the learned parameters and input feature maps, dynamically adjusted convolution kernel weights can be generated.

Dynamic convolution kernel has the following advantages: 1) Strong adaptability: The weight of the convolution kernel is adjusted according to the learned offset so that the network can better adapt to the irregular shape and posture changes of the target. 2) High flexibility: The weight of the convolution kernel can be dynamically adjusted according to the specific conditions of the target, to better capture the characteristic information of the target.

Through the generation of dynamic convolution kernels, the D-DCN network can better adapt to targets of various shapes and postures, thereby improving the accuracy and robustness of target detection. The calculation process of dynamic convolution operation includes the following steps: 1) Position adjustment of the convolution kernel: For each convolution kernel position p, the position of the convolution kernel is dynamically adjusted according to the learned offset $\Delta p_n(\theta, \phi)$. 2) Convolution kernel weight adjustment: Calculate the dynamically adjusted convolution kernel weight $W(p_n, \theta, \phi)$ based on the adjusted convolution kernel position and shape. 3) Calculation of convolution output: For each convolution kernel position p, use the dynamically adjusted convolution kernel weight to perform a weighted sum to obtain the convolution output value $y(p)$. The process can be described as:

$$y(p) = \sum_{p_n \in \mathcal{R}} x(p + p_n + \Delta p_n(\theta, \phi)) \cdot W(p_n, \theta, \phi) \tag{7}$$

Through this operation, the D-DCN network can dynamically adjust the position and weight of the convolution kernel according to the local structure and shape changes of the target, thereby better adapting to the characteristics of the target.

3.4 Loss

When designing the loss function, we hope that it can accurately reflect the difference between the model predictions and the real labels and that it can effectively guide the training process of the model. In target detection tasks, we usually need to consider two aspects of error: positioning error and category classification error.

Position Positioning Error loss function. The location error loss function measures the difference between the model's prediction of the target location and the true location. The commonly used positioning error loss function is Smooth L1 Loss, which is a variant of Huber Loss. It uses square loss for smaller errors and absolute value loss for larger errors to reduce the impact of outliers on training. This loss is expressed

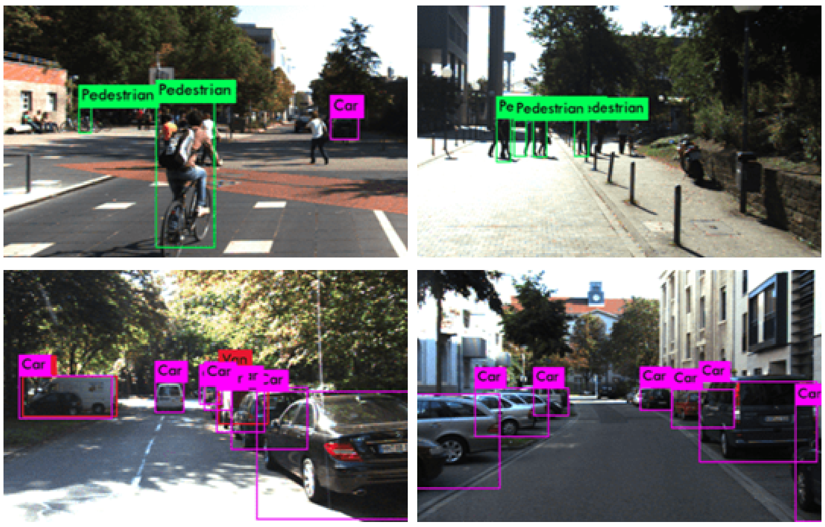


Figure 2. Visualization results on KITTI dataset.

Table 1. Experimental results on the KITTI dataset.

Model	Time	Cars			Pedestrians			Cyclists		
		Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mod	Hard
LSVM-MDPM-sv [9]	10s	68.02	56.48	44.18	47.74	39.36	35.95	35.04	27.50	26.21
DPM-VOC-VP [28]	8s	74.95	64.71	48.76	59.48	44.86	40.37	42.43	31.08	28.23
SubCat [26]	0.7s	84.14	75.46	59.71	54.67	42.34	37.95	-	-	-
3DVP [35]	40s	87.46	75.77	65.38	-	-	-	-	-	-
AOG [19]	3s	84.80	75.94	60.70	-	-	-	-	-	-
Faster-RCNN	2s	86.71	81.84	71.12	78.86	65.90	61.18	72.26	63.35	55.90
CompACT-Deep [2]	1s	-	-	-	70.69	58.74	52.71	-	-	-
DeepParts [33]	1s	-	-	-	70.49	58.67	52.78	-	-	-
FilteredICF [38]	2s	-	-	-	67.65	56.75	51.12	-	-	-
pAUCEnsT [27]	60s	-	-	-	65.26	54.49	48.60	51.62	38.03	33.38
Regionlets [34]	1s	84.75	76.45	59.70	73.14	61.15	55.21	70.41	58.72	51.83
3DOP [3]	3s	90.03	88.64	76.11	81.78	67.47	64.70	78.39	68.94	61.37
SDP+RPN [36]	0.4s	90.14	88.85	78.38	80.09	70.16	64.82	81.37	73.74	65.31
Ours	0.4s	**93.04**	**89.02**	**79.10**	**83.92**	**73.70**	**68.31**	**84.06**	**75.46**	**66.07**

Table 2. Experimental environment.

Parameter	**Configuration**
CPU	Intel Core i9-12700KF
GPU	NVIDIA GeForce RTX A30 (24 GB)
CUDA	CUDA 11.7
Python	Python 3.9.13
Deep learning framework	Pytorch 2.1.0
Operating system	Ubuntu 22.04.2

as:

$$L_{\text{loc}}(l, l^*) = \sum_{i \in \{N\}} \text{SmoothL1}(l_i - l_i^*) \tag{8}$$

where l is the target position parameter predicted by the model, l^* is the real target position parameter, and SmoothL1(x) is the Smooth L1 Loss function.

Classification Error loss function. The classification error loss function measures the difference between the model's prediction of the target class and the true class. A commonly used classification error loss function is the cross-entropy loss function, which imposes a greater penalty on incorrect classification predictions, thereby prompting the model to learn more accurate classifications. The loss function can be expressed as:

$$L_{\text{cls}}(c, c^*) = -\sum_{c'} c'^* \log(c) \tag{9}$$

where c is the target category probability distribution predicted by the model, c^* is the true target category label, and c'^* is the predicted probability corresponding to the true category label.

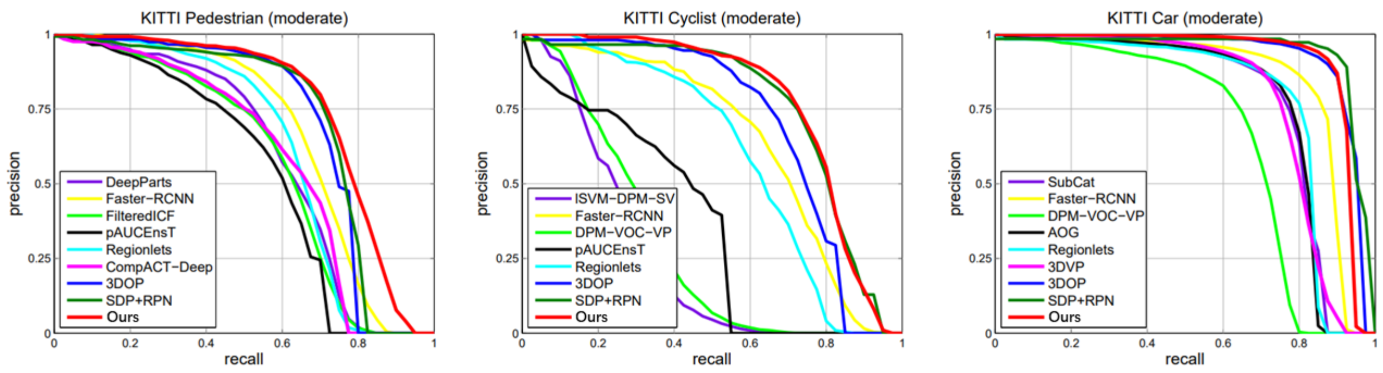


Figure 3. Comparison with state-of-the-art methods on the KITTI dataset.

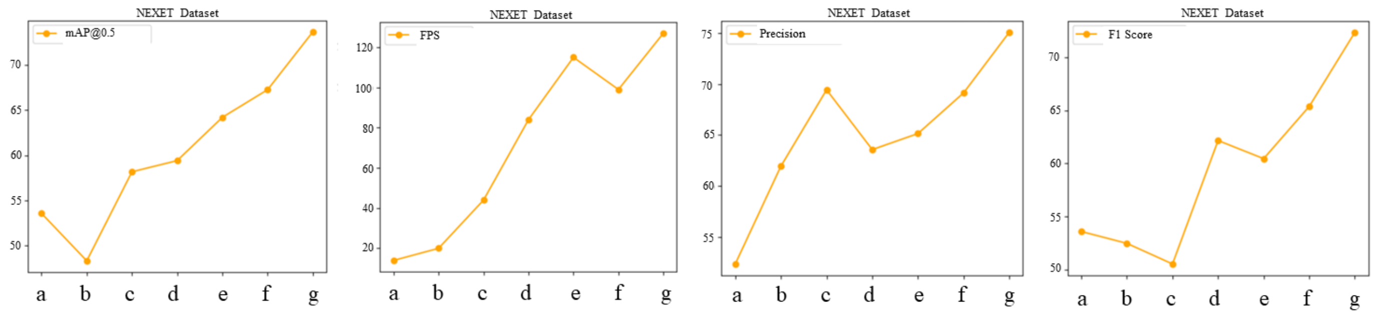


Figure 4. Results on NEXET dataset. a: SubCat; b: 3DVP; c: AOG; d: DeepParts; e: Regionlets; f: FilteredICF; g: Ours.

Table 3. Model Parameter

Parameter	Value
lr	0.0002
Optimizer	Adam
Batch Size	24
Weight Decay	0.0003
Epoch	350
Activation Function	ReLU
Early Stop	True

Comprehensive loss function design. To comprehensively consider positioning error and classification error, we can adopt the form of a joint loss function when designing the loss function, taking into account the errors in these two aspects at the same time. A common approach is to perform a weighted sum of the positioning error loss function and the classification error loss function to obtain a comprehensive loss function:

$$L = L_{loc} + \lambda L_{cls} \tag{10}$$

where λ is the trade-off parameter between positioning error and classification error, which is used to balance the importance of the two in the training process. By designing an appropriate loss function, we can effectively guide the training of the model and improve the performance of the target detection task.

4 Experiments

4.1 Experimental Setting

For a fair comparison, all models were trained on a server equipped with four NVIDIA A30 GPUs. We apply the Adam [18] optimizer with a learning rate of 0.001. Batch Normalization [16] is used following each parameter layer. A weight decay of 0.0001 is used in both networks. See Table 2 and Table 3 for detailed parameters.

4.2 Datasets

KITTI dataset. The KITTI dataset is one of the most popular datasets for use in mobile robotics and autonomous driving. It consists of hours of traffic scenarios recorded with a variety of sensor modalities, including high-resolution RGB, grayscale stereo cameras, and a 3D laser scanner. Despite its popularity, the dataset itself does not contain ground truth for semantic segmentation. However, various researchers have manually annotated parts of the dataset to fit their necessities.

NEXET dataset. The NEXET dataset, a widely utilized video dataset in autonomous driving research, encompasses data from over 77 countries and more

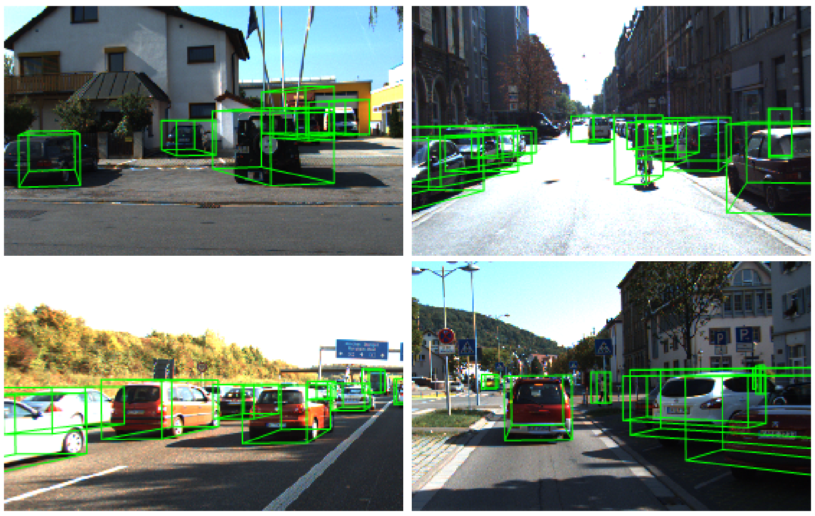


Figure 5. Visualization results on NEXET dataset.

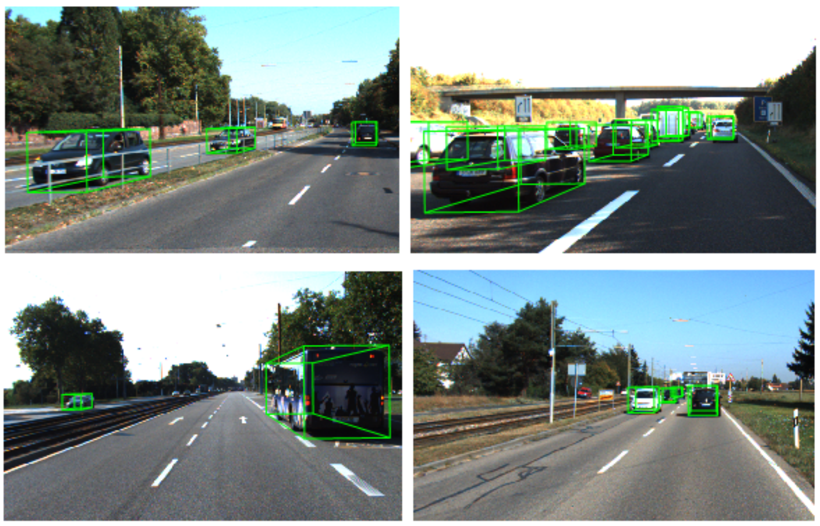


Figure 6. Visualization results on Caltech dataset.

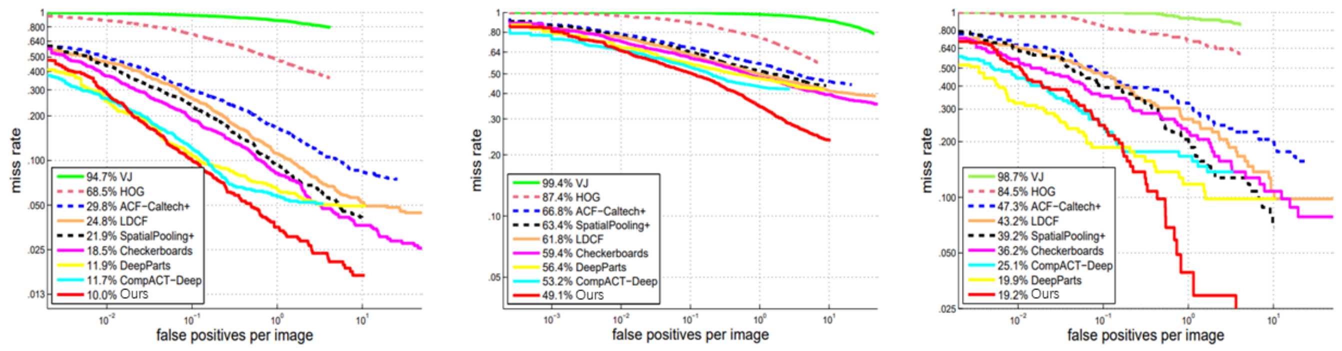
than 1,400 cities. It includes footage captured under three different lighting conditions (day, night, and twilight) and across all four seasons. The dataset also features diverse road types, such as urban, rural, highway, residential, and desert roads, as well as various weather conditions, including clear skies, fog, rain, and snow. With its high-quality video clips and detailed annotations, NEXET serves as an excellent resource for evaluating our method.

Caltech dataset. The Caltech dataset is a widely used dataset for object recognition tasks and contains approximately 9000 images from 101 object categories. The categories were chosen to reflect the variety of objects in the real world, and the images themselves were carefully selected and annotated to provide a challenging benchmark for object recognition algorithms.

Evaluation Methodology. We start with an evaluation of the proposal network. Following literature [15], prediction callbacks are used as a performance metric. To be consistent with KITTI's setting, the ground truth is considered recalled if the intersection over union (IoU) ratio of the best matching proposal is higher than 70% for cars and higher than 50% for pedestrians and cyclists. These benchmarks were chosen because, unlike VOC [7] and ImageNet [31], they contain many small objects. Typical image sizes for KITTI and Caltech are 1250x375 and 640x480 respectively. KITTI contains three categories of objects: cars, pedestrians, and cyclists, and three assessment levels: easy, medium, and hard. The "medium" level is the most commonly used. A total of 7,481 images are available for training/validation, and 7,518 for testing. Since no ground truth is available for

Table 4. Ablation study results. Where "1.5", "2" and "3.5" represent the sampling ratio. "random" and "mixture" represent the sampling method.

Model	Times	params	Cars			Pedestrians		
			Easy	Mod	Hard	Easy	Mod	Hard
1.5	0.12s/0.10s	471M/217M	90.55	87.93	71.90	76.01	69.53	61.57
2	0.43s/0.38s	471M/217M	90.96	88.83	75.19	76.33	72.71	64.31
3.5	0.23s/0.20s	471M/217M	94.08	89.12	75.54	77.74	72.49	64.43
random	0.22s/0.19s	471M/217M	90.94	87.50	71.27	70.69	65.91	58.28
mixture	0.22s/0.19s	471M/217M	90.33	88.12	72.90	75.09	70.49	62.43
IFPN	0.24s/0.20s	352M/191M	92.89	88.88	74.34	76.89	71.45	63.50
D-DCN	0.22s/0.19s	344M/138M	90.49	89.13	74.85	76.82	72.13	64.14
Ours	0.19s/0.18s	103M/82M	82.73	73.49	63.22	64.03	60.54	55.07



(a) reasonable (b) medium (c) partial occlusion

Figure 7. Comparison with state-of-the-art methods on the Caltech dataset.

the test set, we follow the recommendation of [3] and split the training validation set for training and validation. In all reduction experiments, the training set is used for learning and the validation set is used for evaluation. Next, models were trained separately for car detection and pedestrian/bicycle detection. A model for pedestrians was learned on Caltech.

4.3 Results

The results on the KITTI: Comparisons with previous methods are shown in Table 3 and Figure 3. We set a new record for detecting pedestrians and cyclists. Columns "Pedestrians-Mod" and "Cyclists-Mod" are 6 and 7 points higher than 3DOP respectively, and perform better than Faster-RCNN, Regionlets, etc. We also achieved considerable lead on the very new SDP+RPN using scale-dependent pooling. In terms of speed, this network is quite fast. For the largest input size, our detector is approximately 8 times faster than 3DOP. On the original image (1250x375), the detection speed reaches 10fps. Figure 2 shows our visualization results.

The results on the NEXET: Table 5 offers a thorough comparison of the performance of various methods on the NEXET dataset. Our method stands out with impressive metrics, achieving 75.07% in Precision, 72.37% in F1 Score, 73.67% in mAP@0.5, and operating at 127 FPS. These results underscore the reliability and effectiveness of our approach in different scenarios. Additionally, qualitative outcomes presented in Figure 4 and Figure 5 illustrate that our method performs exceptionally well in real-world applications. In summary, our approach has led to significant advancements in both speed and accuracy.

Table 5. Results on the NEXET dataset.

Method	Precision	F1 Score	mAP@0.5	FPS
SubCat	52.36	53.59	53.58	14
3DVP	61.95	52.47	48.33	20
AOG	69.43	50.53	58.18	44
DeepParts	63.57	62.17	59.42	84
Regionlets	65.14	60.46	64.21	115
FilteredICF	69.13	65.39	67.27	99
Ours	**75.07**	**72.37**	**73.67**	**127**

Detection on Caltech: Our detector is also evaluated on the Caltech pedestrian benchmark. The model is compared with methods such as DeepParts, CompACT-Deep, CheckerBoard, LDCF, ACF [5], and SpatialPooling, covering three tasks: Reasonable, medium, and partially obscured. As shown in Figure 7, Our method has state-of-the-art performance. Figure 7 (b) and (c) show that it performs very well on small and occluded objects, surpassing DeepParts which specifically solves the occlusion problem. Furthermore, due to the precision of the proposal network, it misses very few pedestrians. The speed is about 8 fps (15 fps on the Caltech original image 640x480 upsampled to 960x720). Our visualization results are shown in Figure 6.

4.4 Ablation Study

Effect of input upsampling: Table 4 shows that input upsampling can be a key factor in detection. Significant improvements can be achieved by upsampling the input by a factor of 1.5 to 2, but we found that gains beyond 2x are small. This is smaller than the 3.5x factor required by literature [39]. Larger factors cause detection to be significantly slower and require more memory.

Sampling strategies: Table 4 compares the sampling strategies: random and mixture. For cars, the three strategies are similar; for pedestrians, the bootstrap and hybrid strategies are similar, but the random strategy is significantly worse. Note that random sampling has more false positives.

CNN feature approximation: We tried three methods to learn deconvolution layers for feature map approximation: 1) weights with bilinear interpolation; 2) weights initialized by bilinear interpolation and learned by backpropagation; 3) weights initialized by Gaussian noise and learned by backpropagation. We found that the first method works best, which confirms the research results of [17]. As shown in Table 4, deconvolution layers help in most cases. The gains are greater for smaller input images, which tend to contain smaller objects. It is worth noting that the computational complexity of feature map approximation is very small and does not increase parameters.

Object Detection via Proposal Networks: Proposal networks can act as detectors by converting class-agnostic classification into class-specific classification. Table 4 shows that although not as powerful as the unified network, it achieves quite good results, which are better than some detectors on the KITTI rankings.

5 Conclusion

This paper proposes an improved object detection algorithm based on Dynamically Deformable Convolutional Networks (D-DCN), which achieves remarkable results through in-depth study of multi-scale and variability challenges in object detection tasks. Our improved algorithm makes full use of the advantages of deep learning technology, innovates in feature extraction and network structure design, and effectively improves the accuracy and robustness of target detection. In the experimental part, we verified it on two commonly used target detection data sets, KITTI and Caltech. The experimental results show that our algorithm achieved significant performance improvement in the target detection task. Compared with traditional methods, our algorithm performs more robustly and accurately across multiple scales and variability. The improved feature pyramid network can effectively extract multi-scale target features, while the dynamically deformable network can achieve accurate detection based on changes in the shape and posture of the target. The comprehensive loss function effectively guides the training process of the model and improves the generalization ability and robustness of the model.

Overall, our work provides an effective method to solve the multi-scale and variability challenges in target detection tasks, with high practical value and prospects for generalization and application. In the future, we will further explore the application of deep learning technology in the field of target detection, continuously improve algorithm performance, and improve the versatility and practicality of the model.

Conflicts of Interest

The authors declare no conflicts of interest.

Funding

This work was supported without any funding.

References

[1] Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. In *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision*, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9 (pp. 404-417). Springer Berlin Heidelberg. [CrossRef]

[2] Cai, Z., Saberian, M., & Vasconcelos, N. (2015). Learning complexity-aware cascades for deep pedestrian detection. In *Proceedings of the IEEE*

international conference on computer vision (pp. 3361-3369). [CrossRef]

[3] Chen, X., Kundu, K., Zhu, Y., Berneshawi, A. G., Ma, H., Fidler, S., & Urtasun, R. (2015). 3d object proposals for accurate object class detection. *Advances in neural information processing systems*, 28. [CrossRef]

[4] Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., & Wei, Y. (2017). Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 764-773). [CrossRef]

[5] Dollár, P., Appel, R., Belongie, S., & Perona, P. (2014). Fast feature pyramids for object detection. *IEEE transactions on pattern analysis and machine intelligence*, 36(8), 1532-1545. [CrossRef]

[6] Dollar, P., Wojek, C., Schiele, B., & Perona, P. (2011). Pedestrian detection: An evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence*, 34(4), 743-761. [CrossRef]

[7] Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88, 303-338. [CrossRef]

[8] Geiger, A., Lenz, P., & Urtasun, R. (2012, June). Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition* (pp. 3354-3361). IEEE. [CrossRef]

[9] Geiger, A., Wojek, C., & Urtasun, R. (2011). Joint 3d estimation of objects and scene layout. *Advances in Neural Information Processing Systems*, 24.

[10] Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440-1448). [CrossRef]

[11] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587). [CrossRef]

[12] Glorot, X., Bordes, A., & Bengio, Y. (2011, June). Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics* (pp. 315-323). JMLR Workshop and Conference Proceedings. [CrossRef]

[13] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 2961-2969). [CrossRef]

[14] Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4), 18-28. [CrossRef]

[15] Hosang, J., Benenson, R., Dollár, P., & Schiele, B. (2015). What makes for effective detection proposals?. *IEEE transactions on pattern analysis and machine intelligence*, 38(4), 814-830. [CrossRef]

[16] Ioffe, S., & Szegedy, C. (2015, June). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448-456). pmlr. [CrossRef]

[17] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., ... & Darrell, T. (2014, November). Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia* (pp. 675-678). [CrossRef]

[18] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint* arXiv:1412.6980. [CrossRef]

[19] Li, B., Wu, T., & Zhu, S. C. (2014). Integrating context and occlusion for car detection by hierarchical and-or model. In *Computer Vision–ECCV 2014: 13th European Conference*, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13 (pp. 652-667). Springer International Publishing. [CrossRef]

[20] Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2117-2125). [CrossRef]

[21] Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 2980-2988). [CrossRef]

[22] Cruz-Mota, J., Bogdanova, I., Paquier, B., Bierlaire, M., & Thiran, J. P. (2012). Scale invariant feature transform on the sphere: Theory and applications. International journal of computer vision, 98, 217-241. [CrossRef]

[23] Cheng, L., Wang, Y., Liu, Q., Epema, D. H., Liu, C., Mao, Y., & Murphy, J. (2021). Network-aware locality scheduling for distributed data operators in data centers. *IEEE Transactions on Parallel and Distributed Systems*, 32(6), 1494-1510. [CrossRef]

[24] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference*, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14 (pp. 21-37). Springer International Publishing. [CrossRef]

[25] Nezamabadi-pour, H., & Kabir, E. (2004). Image retrieval using histograms of uni-color and bi-color blocks and directional changes in intensity gradient. *Pattern Recognition Letters*, 25(14), 1547-1557. [CrossRef]

[26] Ohn-Bar, E., & Trivedi, M. M. (2015). Learning to detect vehicles by clustering appearance patterns. *IEEE Transactions on Intelligent Transportation Systems*, 16(5), 2511-2521. [CrossRef]

[27] Paisitkriangkrai, S., Shen, C., & van den Hengel, A. (2015). Pedestrian detection with spatially pooled features and structured ensemble learning. *IEEE transactions on pattern analysis and machine intelligence*,

38(6), 1243-1257. [CrossRef]

[28] Pepik, B., Stark, M., Gehler, P., & Schiele, B. (2015). Multi-view and 3d deformable part models. *IEEE transactions on pattern analysis and machine intelligence*, 37(11), 2232-2245. [CrossRef]

[29] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788). [CrossRef]

[30] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

[31] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115, 211-252. [CrossRef]

[32] Tan, M., Pang, R., & Le, Q. V. (2020). Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 10781-10790).

[33] Tian, Y., Luo, P., Wang, X., & Tang, X. (2015). Deep learning strong parts for pedestrian detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 1904-1912). [CrossRef]

[34] Wang, X., Yang, M., Zhu, S., & Lin, Y. (2013). Regionlets for generic object detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 17-24). [CrossRef]

[35] Xiang, Y., Choi, W., Lin, Y., & Savarese, S. (2015). Data-driven 3d voxel patterns for object category recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1903-1911). [CrossRef]

[36] Yang, F., Choi, W., & Lin, Y. (2016). Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2129-2137). [CrossRef]

[37] Wang, C., Wang, Y., Han, Y., Song, L., Quan, Z., Li, J., & Li, X. (2017, January). CNN-based object detection solutions for embedded heterogeneous multicore SoCs. In *2017 22nd Asia and South Pacific design automation conference (ASP-DAC)* (pp. 105-110). IEEE. [CrossRef]

[38] Zhang, S., Benenson, R., & Schiele, B. (2015, June). Filtered channel features for pedestrian detection. In *CVPR* (Vol. 1, No. 2, p. 4).

[39] Zhu, Y., Urtasun, R., Salakhutdinov, R., & Fidler, S. (2015). segdeepm: Exploiting segmentation and context in deep neural networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4703-4711). [CrossRef]