New Materials, Machinery and Vehicle Engineering
J. Xu and J.P. Davim (Eds.)

doi:10.3233/ATDE230123

DH-Net: Dynamic Head for Object Detection Network

Taojun DING[a], Xi ZHANG[a,1], Chuan HU[a], Yonghang SHAN[a], Baixuan ZHAO[a], Yiwei LIN[a] and Shuang HU[a]
[a] *School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China*

Abstract. Object detection performs a vital function in numerous domains. In contemporary object detection frameworks, multiple detection heads are a crucial element. Multi-performance head improvement is mostly attributable to the employment of many heads to detect objects of different sizes, but this also results in significant computing usage. Moreover, not all heads need to participate in the detecting process for every image. We first investigate the correlation between the number of heads and computing consumption, explore how to determine whether a detection head is in charge of detecting an object, and propose a dynamic head network for detection (DH-Net), which according to the input photographs, adaptively pick which heads to make predictions. As a result, the technique can accomplish more effective dynamic reasoning and a better balance between detection accuracy and computing expense. DH-Net allows the model to lower the number of parameters by up to 34% while still maintaining comparable accuracy, according to a series of controlled trials in the TT100K datasets. Moreover, it has the effect of speeding up training for redundant models.

Keywords. Deep learning, CNN, object detection, dynamic neural network

1. Introduction

Deep learning-based object detection[3][2] networks are a popular issue in the area of computer vision. The development of CNN has substantially increased the accuracy of object identification, which plays an essential role in several industries, including autonomous vehicles, video surveillance systems, medical imaging, and industrial production quality monitoring.

The majority of the currently used object detection networks are static networks[3][2], meaning that the inference process fixes the computation graph and network parameters. Detectors are separated into one-stage and two-stage detectors depending on whether there is an explicit proposal generation step. The one-stage approach predicts the item category and position by immediately extracting characteristics from the input picture. The YOLO series[3], and RetinaNet[7] are the most emblematic. The most typical two-stage technique uses the R-CNN series[2] to produce candidate frames using the RPN or Anchor method firstly and then optimizes the generated candidate frames secondly to acquire more precise results. These static networks handle all samples equally, and to get the final detection result, each sample must pass through the whole model. Research[4][5] has shown that the whole model is

[1] Corresponding Author. Xi ZHANG, Shanghai Jiao Tong University, 800 Dongchuan Road, Minhang District, Shanghai, China; E-mail address: braver1980@sjtu.edu.cn.

unnecessary for "simple" data. And the deployment of big and high-performance models in devices with limited processing capacity is somewhat constrained as a result of the fixed computing power consumption of redundant models.

The dynamic neural network can effectively resolve the issue. Unlike the static neural network, the model is adaptively adjusted during inference based on the input samples. Recent research on dynamic networks has demonstrated that for straightforward, normative samples, the model can accomplish accurate and rapid inference using only a portion of the network, while only a small number of non-normal samples require the use of the entire network. According to the input, the adaptive adjustment model may alter the structure or parameters of the model. For the problem of image classification, BranchyNet[5] proposes a confidence-based early departure mechanism. When the network prediction confidence exceeds the threshold, it will exit early. Bolukbasi[6] et al. use various-sized models to obtain inference using a learnable decision function. SkipNet[4] targets the numerous repeated block structures in resnet, and a parallel gate is connected in each block to control whether the block is bypassed. In the field of object detection, by adaptively deciding if an image is easy or difficult and choosing the suitable detector for it, Zhou[8] et al. combined fast (but less accurate) detectors with accurate (but slower) detectors. In the DyFPN[1] model proposed by Zhu et al., the parameter quantity is used as a component of the loss, and convolution kernels in FPN can be obtained by learning the opening and closing of gates.

In contrast to previous dynamic neural network research, DH-Net focuses on balancing the computational demands made by the multi-head target detector with the detection accuracy it brings. Although more heads typically increase accuracy, they also necessitate more calculations, and not all heads must be used for every image in order to obtain accurate results. The learnable gate allows the DH-Net to adaptively choose the number of heads to utilize based on the input image, which can decrease the number of parameters by up to 34% and the gate prediction accuracy to over 92%. The following are the contributions of the paper:

(1) Generic Dynamic Detection Head: Aiming at the problem of massive calculation consumption caused by multi-head, a dynamic number detection head strategy is proposed, which achieves the effect of almost no loss of accuracy, but greatly reduces the number of parameters.

(2) One dynamic model for multiple scenes: Aiming at the problem of training models of different sizes under different equipment conditions, a method of adjusting the average network size by manually adjusting the threshold is proposed, which achieves the effect of training a model, adapting to various computing power equipment and adjusting the inference speed.

2. Method

Overview: The objective is to enable the model to alter the number of object detection heads adaptively through learning to achieve faster and more precise detection.

The strategy is to extract labels from each image during training; these labels indicate whether the image contains objects of certain sizes. These labels are learned by some tiny branches added to the network. During inference, based on the output of these branches, the model determines whether to open the other branches. Figure 1 shows two cases where different numbers of heads are required. Next, we'll go through the pipeline

of the network, how to determine the target size, the layout of the gating component, and the loss of the model.

Figure 1. Two cases where different numbers of heads are required, the left needs one head, right needs three heads.

2.1. Overall network pipeline

Figure 2 depicts the whole pipeline. There are three distinct components to the current detectors. One or more level characteristics of the image are first extracted using the backbone. These traits are combined in the neck. The head also forecasts the bounding boxes and item categorization. Several heads are created to be in charge of bounding boxes (bboxes) of various ranges to prevent over-coupling bboxes of various sizes.

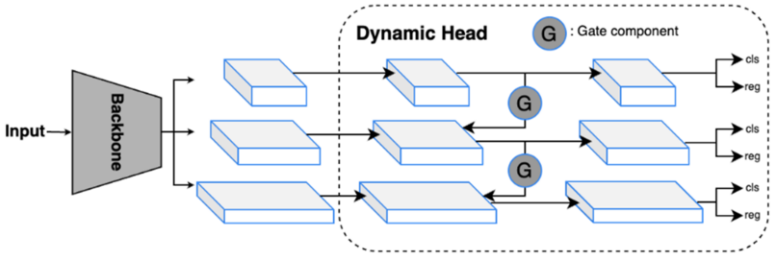

Figure 2. Overall network pipeline.

To choose a few heads for prediction, DH-Net monitors the opening and shutting of the gate using the labels acquired from the image. Specifically, given that x^i is the input of the i-th head and the input of the i-th gate, the output $y^i$of the current head and the input x^{i+1} of the next head as follows:

$$y^i = \begin{cases} None, \ if \ \ x^i = None \\ H^i(x^i), \ otherwise \end{cases} \tag{1}$$

$$x^{i+1} = G^i(x^i)F^i(x^i) \tag{2}$$

where H is a head detector, and $G(x) \in \{0,1\}$ is a gating function.

DH-Net offers distinct training and testing procedures. Regardless of the output of the gating equation, all heads will be used throughout training, and joint supervisory training will be conducted on the output of all heads and gates. In contrast, while

predicting, if the output of the gate is below a certain threshold, it will quit early and use the existing head output to obtain the final detection result.

2.2. Object size definition

How to determine the size of the target is the fundamental concept of DH-Net, which will be used to supervise whether the model exits from the current head. Initially, the range of sizes for which a head is accountable is described. Assuming that the i-th head preset box is $h_j^i(i = 1,2,3\ldots N, j = 1,2,3\ldots M)$ in height and w_j^i in width, and that the k-th item in the image is w^k(k = 1,2,3…P) in height and h^k, the following definition determines whether the head is in charge of detecting this object:

$$r_{w,j,k}^i = max\left(\frac{w^k}{w_j^i}, \frac{w_j^i}{w^k}\right) \tag{3}$$

$$r_{h,j,k}^i = max\left(\frac{h^k}{h_j^i}, \frac{h_j^i}{h^k}\right) \tag{4}$$

$$\theta_{j,k}^i = r_{w,j}^i < \varepsilon \wedge r_{h,j}^i < \varepsilon \tag{5}$$

$$\varphi_k^i = \bigvee_{j=1}^M \theta_j^i \tag{6}$$

where ε is a hyperparameter, and if φ^i is 0, It indicates that the head is not required to be accountable for the object; otherwise, it is responsible. Second, as long as there is a box in the current head preset boxes that is accountable for the item, the head is regarded to require to detect the object, and φ^i is given the value 1.

Then define whether the head is responsible for detecting this picture:

$$\phi^i = \left(\bigvee_{k=1}^P \varphi_k^i\right) \vee \phi^{i+1} \tag{7}$$

If ϕ^i is 0, it means that the head does not need to be responsible for the picture, and vice versa. And ϕ^{N+1} is set to 0. The result thus obtained can be used as a label for training gates.

2.3. Gating Component Design

As illustrated in Figure 3, the dynamic gate comprises two convolutional layers, two fully linked layers(MLP), and an activation function that generates switching signals. On the one hand, the two convolutional layers, including batch normalization and SiLU, rapidly decrease the channels of the feature, preventing the direct usage of the fully connected layer from introducing too many new parameters. On the other hand, it can limit the effect of the gate branch on the original network. The two MLPs further lower the size of the feature vector, and the second MLP outputs a one-dimensional vector α^i. A gate signal may be generated by feeding this one-dimensional vector into an activation function, such as a sigmoid. Sigmoid can generate a signal between 0 and 1. After the model has been trained, the threshold can be modified to increase the flexibility of the model by balancing its computational power consumption and accuracy rate.

Figure 3. Structure of Gating Components.

2.4. Loss

The loss function extends the conventional object identification loss by accounting for the difference between the predicted gate signal and the ground-truth labels derived from the picture. The object detection loss function is shown in Equation (8), including confidence loss L_{conf}, classification loss, and bounding box regression loss L_{box}, where λ_{conf}, λ_{cls} and λ_{box} are their weights, $I_{i,j}^{obj}$ denotes anchor frames with targets, $I_{i,j}^{nobj}$ denotes anchor frames without marks. IOU is the intersection-over-union ratio between two bounding boxes, $\rho^2(b, b^{gt})$is the distance between the center of the bounding box.

$$L_{det} = \lambda_{conf} L_{conf} + \lambda_{cls} L_{cls} + \lambda_{box} L_{box} \tag{8}$$

$$L_{conf} = -\sum_{i=0}^{S\times S} I_{i,j}^{obj} \left[\hat{c}_i^j \, log\left(c_i^j\right) + \left(1-\hat{c}_i^j\right) log\left(1-c_i^j\right)\right] - \lambda_{noobj} \sum_{i=0}^{S\times S} \sum_{j=0}^{M} I_{i,j}^{nobj} \left[\hat{c}_i^j \, log\left(c_i^j\right) + \left(1-\hat{c}_i^j\right) log\left(1-c_i^j\right)\right] \tag{9}$$

$$L_{cls} = -\sum_{i=0}^{S\times s} I_{ij}^{obj} \sum_{c\in classes} \left[\hat{p}_i^j \, log\left(p_i^j\right) + \left(1-\hat{p}_i^j\right) log\left(1-p_i^j\right)\right] \tag{10}$$

$$L_{box} = 1 - IOU + \frac{\rho^2(b,b^{gt})}{c^2} + \alpha v \tag{11}$$

$$v = \frac{4}{\pi^2}\left(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w}{h}\right)^2 \tag{12}$$

$$\alpha = \frac{v}{(1-IOU)+v} \tag{13}$$

The prediction gate loss calculation method is shown in Equation (14). The $\lambda_{pos,i}$ (representing the ratio of each head on and off) is introduced to balance the backpropagation gradient ratio caused by the highly unbalanced loss of positive and negative samples to achieve a higher effect[7].

$$L_{gate} = \sum_{i=0}^{H} \left[\lambda_{pos,i} \hat{g}_i \, log(g_i) + (1-\hat{g}_i) \, log(1-g_i)\right] \tag{14}$$

The following is the total loss calculation formula:

$$L = L_{det} + \lambda L_{gate} \tag{15}$$

The λ is introduced to achieve a compromise between the precision of model recognition and the accuracy of forecasting gates.

3. Experiments and Results

3.1. Dataset

All of the experiments are trained and evaluated using TT100k datasets. Given that the number of images in specific categories of the datasets is relatively low, the categories with less than 15 images during preprocessing were removed, leaving 33 categories in the remaining datasets. Table 1 displays the number of images in the train set and test set with large, medium, and small targets, as defined by the approach described before.

Table 1. The distribution of large, medium, and small objects in the dataset.

Datasets	All	Large	Medium	Small
Train datasets	5840	1918	4550	5802
Test datasets	2941	926	2304	2909

3.2. Evaluation Metrics

The signal quality of gate prediction is assessed using Accuracy and Recall, on the one hand. The accuracy rate is used to assess if the predicted outcomes of the model are accurate, and the computation formula is as follows:

$$Acc = \frac{TP+TN}{TP+FP+TN+FN} \tag{16}$$

Among them, Acc is the Accuracy rate. T (true), F (false), P (positive), and N (negative) are abbreviations.

Using recall to evaluate the capability of the model to identify positive samples. The proportion of missed positive samples decreases with increasing recall. The following is the calculating formula:

$$R = \frac{TP}{TP+FN} \tag{17}$$

On the other hand, AP (Average Precision) is often used to assess the detection performance in the area of object detection. Here is the formula for calculating:

$$P = \frac{TP}{TP+FP} \tag{18}$$

$$AP = \int_0^1 P(R)dR \tag{19}$$

3.3. Implementation Details

For all experiments, all models of the Yolov5 series were end-to-end trained on a machine with 2 NVIDIA RTX 3090 GPUs. Training each model for 300 epochs with the SGD optimizer. The learning rate is warmed up at the start of the training by linearly

raising it for 50 epochs, and then using the cosine annealing procedure, it is progressively decreased from 0.01 to 0.001. The input picture resolution is 1024×1024, and each batch size increases as much as feasible to fully use the video memory. In contrast, the batch size for each group of control experiments remains the same. The $\lambda_{pos,i}$ of the two heads in Equation (14) are set to 0.28 and 2.0, respectively, the loss weight λ of the predicted gate signal is set to 0.2, and ε is set to 4. Also, mosaic[3] was disabled to prevent modifications to the sample distribution brought on by random mosaic.

3.4. Main Results

3.4.1. Effectiveness and Efficiency of DH-Net

Table 2 shows the reduction in parameters and calculation when the model opens just one or two heads. Evidently, the computation and parameter count of model, particularly the parameter count, have decreased.

Table 2. Decreases and percentages of parameter volume and FLOPs volume.

Model	One-head		Two-head	
	Params	FLOPs	Params	FLOPs
DH-Yolov5n	-0.65M(-34.7%)	-1.06G(-19.0%)	-0.47M(-25.4%)	-0.48G(-8.6%)
DH-Yolov5s	-2.49M(-34.0%)	-4.09G(-19.4%)	-1.83M(-24.9%)	-1.87G(-8.9%)
DH-Yolov5m	-7.39M(-34.3%)	-12.1G(-19.2%)	-5.55M(-25.8%)	-5.68G(-9.0%)

The performance of the original model of yolov5 with different sizes and the model of DH-Net with dynamic head turned on were compared in Figure 4. The structure of Yolov5n, Yolov5s, and Yolov5m is roughly the same, but the depth and width of the model gradually increase. As observed, the DH-Net model predicts the gate signal with increasing accuracy as the model size increases. The detection performance also improves, even outperforming the original model.

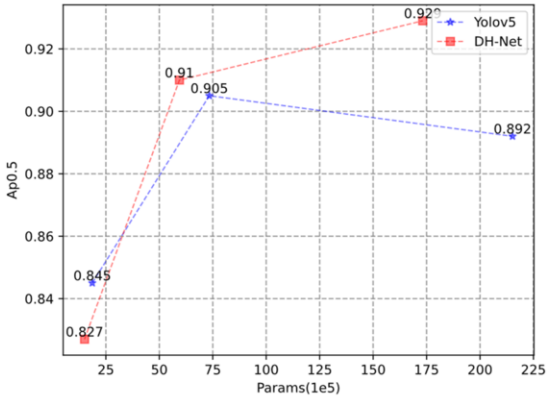

Figure 4. Comparison of parameters and performance between DH-Net and the original model.

The performance of the two heads in DH-Net is shown in Figure 5.

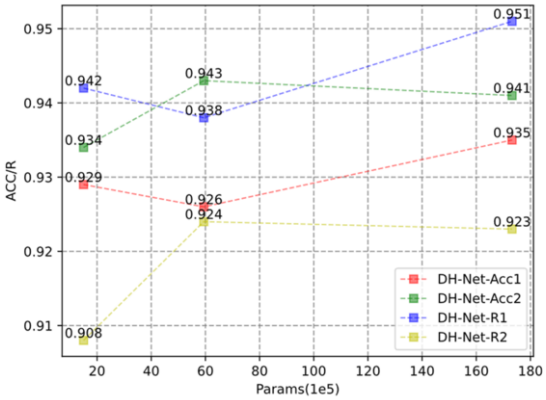

Figure 5. Acc and Recall of two heads.

3.4.2. Accelerating redundant model training

Figure 6 depicts how the detection performance of models varies with training epochs using DH-Yolov5m and original Yolov5m. Several studies on deep learning have demonstrated that bigger models and deeper networks often lead to better outcomes. Still, the research shows that when the model is too huge, the effectiveness of the original model declines. This problem is caused by training difficulty due to too many parameters. According to the experiments, supervision of the gate signal helps decouple the model and hasten model convergence.

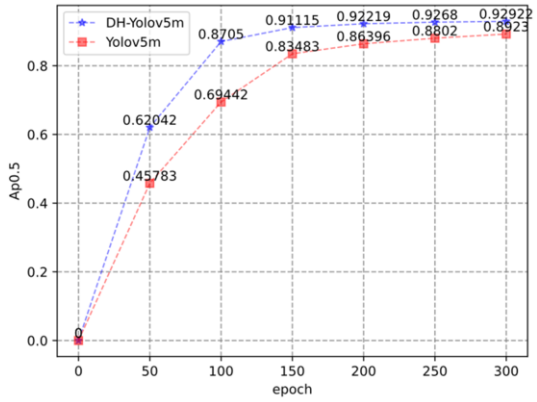

Figure 6. The detection performance of the model changes with epoch.

3.4.3. Balance inference performance and speed

The balance between employing fewer heads and obtaining a higher AP can be artificially adjusted by changing the amount of thresh. This allows the model to adapt to different situations after training, such as increasing the thresh threshold to cope with insufficient computing resources and enhance detection speed. In situations with ample computing capacity, reducing thresh can accomplish the objective of enhancing detection precision. Figure 7 shows the AP and time-consuming changes of the model when adjusting the first thresh value.

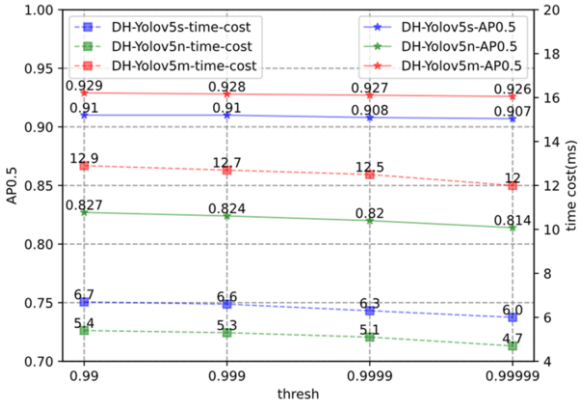

Figure 7. AP and time-consuming changes with thresh.

4. Conclusions

In this research, we propose a dynamic head for object detection that adaptively tunes the number of heads in accordance with the input image and provides a superior trade-off between computer power consumption and detection accuracy. Compared with the static, dynamic object detection network, it significantly reduces the number of parameters and floating-point calculations in the inference process, adding almost negligible computational cost. By simply adding several gate components and raising the supervision loss, the dynamic head structure can be quickly and easily included in the already-existing multi-head object detection network. After the training, the accuracy and detection speed may be manually modified.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under 52177218.

References

[1] Zhu M, Han K, Yu C, et al. Dynamic feature pyramid networks for object detection[J]. arXiv preprint arXiv:2012.00779, 2020.
[2] Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[J]. Advances in neural information processing systems, 2015, 28.
[3] Bochkovskiy A, Wang C Y, Liao H Y M. Yolov4: Optimal speed and accuracy of object detection[J]. arXiv preprint arXiv:2004.10934, 2020.
[4] Harvey N J A, Dunagan J, Jones M, et al. Skipnet: A scalable overlay network with practical locality properties[J]. 2002.
[5] Teerapittayanon S, McDanel B, Kung H T. Branchynet: Fast inference via early exiting from deep neural networks[C]//2016 23rd International Conference on Pattern Recognition (ICPR). IEEE, 2016: 2464-2469.
[6] Bolukbasi T, Wang J, Dekel O, et al. Adaptive neural networks for efficient inference[C]//International Conference on Machine Learning. PMLR, 2017: 527-536.

[7] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection[C]//Proceedings of the IEEE international conference on computer vision. 2017: 2980-2988.
[8] Zhou H Y, Gao B B, Wu J. Adaptive feeding: Achieving fast and accurate detections by adaptively combining object detectors[C]//Proceedings of the IEEE International Conference on Computer Vision. 2017: 3505-3513.