Journal of Computer Science and Cybernetics, V.38, N.2 (2022), 131–145
DOI no 10.15625/1813-9663/38/2/16252

DLAFS CASCADE R-CNN: AN OBJECT DETECTOR BASED ON DYNAMIC LABEL ASSIGNMENT

DOANH BUI CAO[1], NGUYEN VO DUY[1], KHANG NGUYEN[1,*]

[1] *University of Information Technology, Vietnam National University, Ho Chi Minh City, Vietnam*

Abstract. Object detection based on Deep Learning is the revolution of computer science in general and related problems of object detection in particular. In particular, recently, two-stage or multi-stage methods of the R-CNN family have shown outstanding results. These methods have two steps in common: Generating proposal boxes and object classification. In the step of the generating proposal, a Regional Proposal Network (RPN) will be learned to suggest high probability regions in the image, and the part of Label Assignment for RPN is of great interest. If the samples are obtained well, RPN will learn well and help the efficiency of the next stage increase sharply. In this study, we investigate and study to improve the object detection performance when applying Dynamic Label Assignment on the first stage of Cascade R-CNN called DLAFS Cascade R-CNN and perform some experiments to prove the effectiveness. Our DLAFS Cascade R-CNN outperform previous methods on three datasets: SeaShips (**+0.2% AP**), UIT-DODV (**+5.7% AP**), MS-COCO (**+2.8% AP**).

Keywords. Object detection; Marine vehicle; Cascade R-CNN; Dynamic training; Document detection.

1. INTRODUCTION

Object detection methods can be divided into two groups: two-stage and one-stage detectors. Two-stage detectors, especially R-CNN family methods, have been showing the outperforming result. Many previous studies surveyed and applied these two-stage methods on different data domains and achieved high results: In [19] surveyed detectors' performance on document data; [11] do experiments on data captured by drones. One-stage detectors are less accurate but faster than two-stage detectors so that they can satisfy the real-time requirements in real life [10]. In this study, we focus on two-stage methods for improvement. The general feature of these two-stage detectors is two stages of processing: 1) generating proposal boxes that may contain objects; 2) refining these boxes and predicting the class of objects inside proposal boxes. In generating proposals step, a Regional Proposal Network (RPN) is trained to generate proposal boxes. Its learning samples come from predefined anchor boxes; anchor boxes are specified as negative (no object) and positive (object) with

*Corresponding author.
E-mail addresses: 19521366@gm.uit.edu.vn (D.C.Bui); nguyenvd@uit.edu.vn (N.D.Vo); khangnttm@uit.edu.vn (K.Nguyen).

offsets which are defined as $(\delta x, \delta y, \delta w, \delta h)$. However, the way of choosing positive and negative samples is relatively unclear. Commonly, we select anchor boxes as positive samples when they have IoU values with their ground-truth higher than I_+, and the anchor boxes which have IoU less than I_- are selected as negative samples. Two IoU thresholds I_+ and I_- are predefined in the training configuration. In the study [21], the authors indicate that this way of selecting samples is not suitable because we will easily ignore some good samples (e.g., hard negative samples) when the selecting process depends on constant values. Therefore, it is necessary to change I_+ and I_- appropriately based on the proposal boxes' quality statistics generated from the RPN.

In this study, we contribute three main points: propose DLAFS Cascade R-CNN that apply Dynamic Label Assignment to adjust IoU threshold based on statistics of proposal boxes on multi-stage detector Cascade R-CNN on the first stage; do experiments of changing hyper-parameters to observe the performance; compare and evaluate on three datasets: SeaShips, UIT-DODV and MS-COCO for proving the improvement of our proposed method on different domains. With the SeaShips and UIT-DODV dataset, we compare the performance with and without applying Dynamic Label Assignment on Faster R-CNN and Cascade R-CNN. With MS-COCO, we directly compare with results in [21] on the test-dev set. It should be clearly noted that we are only focusing on Label Assignment to better select samples for RPN, and we do not impact the process of refining coordinates.

The rest of the paper is structured as follows. Section 2 provides an overview of related work. Section 3 describes how we apply dynamic training technique to our research. Section 4 shows our experiments and discussions. Finally, conclusions are drawn in Section 5.

2. RELATED WORKS

2.1. Object detection methods

In this section, we present an overview of object detection methods and two classical studies about object detection: Faster R - CNN and Cascade R - CNN.

2.1.1. Previous object detection methods

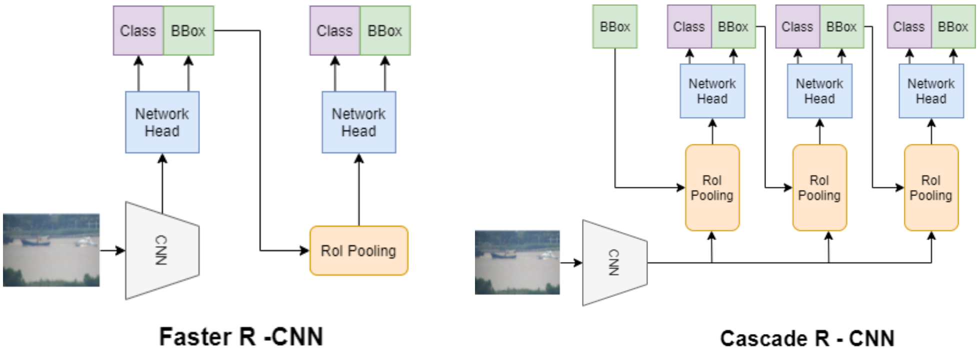
Most of the current object detection methods are divided into two categories: Single-stage and two-stage. The main idea of the two-stage methods: 1) Extract proposed regions that may or may not contain objects; 2) Classifying these proposed regions. Two-stage methods can be mentioned such as R-CNN [5], Faster R-CNN [16], Cascade R-CNN [2], Mask R-CNN [6], Libra R-CNN [12]. The two-stage methods have high accuracy but do not meet the real-time applicability. The one-stage methods make predictions with respect to anchors or a grid of possible object centres. One-stage methods include the YOLO family [1, 13, 14, 15], SSD [9], or more recently we have seen the appearance of CornerNet [7], an object detection method that uses a pair of key points of left-top, right-bottom using a single CNN architecture. CenterNet [4] is followed by CornerNet; instead of using a pair of key points, it uses a set of three points for an object. Single-stage methods often achieve real-time, practical applicability.

2.1.2. Faster R - CNN

In 2014, Ren et al. introduced the Faster R - CNN method, specifically, the authors introduced the concept of Region proposal to create regions with the ability to contain objects based on the selective search algorithm. These regions are then scaled to the same size and further traversed through a CNN architecture. At this step, the proposed regions with threshold $IoU \geq 0.5$ with ground truths will be further predicted by a classifier, and the object's coordinates are predicted by the Bounding box regressor.

2.1.3. Cascade R - CNN

Cascade R – CNN is an object detection method proposed by Cai et al. to solve the problem of decreasing detection performance when the IoU threshold increases. Two main factors influence this: 1) The overmatch occurs when the positive suggested regions disappear exponentially. 2) The prediction time mismatch between the optimal IoU thresholds of the detector and the IoUs of the region proposals. The authors proposed to expand R - CNN into several stages; after each stage, the results will be better selected to eliminate false-positive regions.



(a) Faster R - CNN (b) Cascade R - CNN

Figure 1: A comparison between Faster R - CNN and Cascade R - CNN

As can be seen in Figure 1, Faster R - CNN includes a Region Proposal Network (denoted as the first "Network head" in the figure) to produce object proposals. These proposals are continued to be processed and classified by an RoI detection sub-network (denoted as the second "Network head"). Finally, each region proposal's classification score and bounding box coordinate will be assigned. But in Cascade study [2], the authors claim that these results are not good enough. Thereby, Cai and Vasconelos proposed Cascade R - CNN to handle limitations that exist in Faster R - CNN. In detail, predicted bounding boxes will be input into the network head at the previous stage and continue to go through the next stage to regress bounding boxes again. Equation 1 below shows the operation of the Cascade R-CNN method

$$f(x, \boldsymbol{b}) = f_T \circ f_{T-1} \circ \cdots \circ f_1(x, \boldsymbol{b}). \qquad (1)$$

This iterative approach attempts to gradually fine-tune the bounding box to obtain a more accurate one; in this way, the region proposals are improved progressively. In addition, network heads are different at each stage in Cascade R - CNN and the IoU threshold are changed from small to large through stages. It is worth notable that cascaded regression is a resampling procedure, not a post-processing step, providing good positive samples to the next stage.

2.2. Proposal classification in R - CNN-based detectors

In R-CNN-based detectors, they use a pre-defined IoU threshold for proposal classification, and this task can be formulated as follows

$$Label = \begin{cases} 1, & \text{if } \; maxIoU(b, G) \geq T_+ \\ 0, & \text{if } \; maxIoU(b, G) < T_- \\ -1, & \text{otherwise,} \end{cases}$$

where, b stands for a single proposal; G presents a set of ground truths; T_+ and T_- are the positive and negative thresholds for IoU; 1, 0, -1 stand for positives, negatives and ignored samples, respectively.

The simplest idea to improve the performance is that we can increase the IoU threshold. But at the beginning of the training process, the quality of proposals is not good enough to satisfy the high IoU, so the number of positive proposals can be much low. In Cascade R-CNN, the authors try to handle this issue by increasing IoU threshold T_+ and T_- through the stages with the hope that proposals in previous stages will have acceptable quality for the increased IoU threshold, which is effective but time-consuming [21].

2.3. Previous label assignment optimization methods

As discussed in Section 1, the improvement of the Label Assignment is necessary because if only one IoU threshold is applied over the entire training process, it will be easy to ignore the good samples to let the RPN learn well. There has been a lot of research related to improving this problem in many ways, including the idea of Dynamic Training. GuidedAnchoring [20] proposes a method to predict the locations where the midpoints of the proposal boxes may exist, as well as the sizes and scales at different locations. ATSS [22] proposes an adaptive labeling method. They will initially select the anchor boxes based on the distance between the anchor boxes and the ground truth. Then they calculate the median and the IoU variance of the anchor boxes. Anchor boxes with IoU greater than the sum of the anchor boxes will be selected as positive samples. This study has been experimented with and shows improvements. However, the authors require additional layers and complex structures or only one anchor box to have a full classification score. This is not appropriate in cases where there are many high-quality and high-competition anchors. Our study is heavily inspired by Dynamic Training [21], which shows that the proposed regions generated by RPN get better with training time both in terms of classification and coordinate regression. However, the authors only performed experiments on Faster R-CNN, and the selected hyperparameters were still emotional.

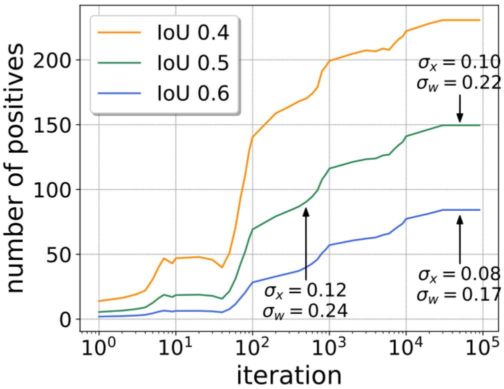


Figure 2: The number of positive proposals under different IoU thresholds during the training process. The curve shows the numbers of positives vary significantly during training, with corresponding changes in regression label distribution [21].

3. METHOD

3.1. Motivation

In the study [21], the authors mention that it is not clear how to assign positive and negative labels because their separation may be ambiguous. The common strategy is to set an IoU threshold of proposals and corresponding ground truth. The authors also indicate the fact it is overlooked that the quality of proposals is indeed improved during training on Figure 2, the number of positives still increases significantly even under different IoU thresholds.

By observation, they proposed the Dynamic Training approach includes two components: Dynamic Label Assignment and Dynamic SmoothL1 Loss for classification and regression branches. In detail, they set the threshold as the IoU of the proposal at a certain percentage since it can reflect the quality of the overall distribution in a specific iteration. For the regression task, they change the shape of the regression loss function to adaptively fit the distribution change of the regression label and ensure the contribution of high-quality proposals to training. In detail, they adjust the β in SmoothL1 loss based on the regression label distribution because β actually controls the magnitude of the gradient of small errors. But in our study, we only use the idea of Dynamic Label Assignment and apply it to Faster R-CNN and Cascade R-CNN.

3.2. Adjusting IoU using Dynamic Label Assignment

In study [21], inspired by the observation that the positive proposals significantly increase in spite of different IoU thresholds, the authors proposed a method to change the values of T_+, T_- according to the statistic of proposals through iterations. The threshold will be updated again after C iterations based on the statistic of the proposed regions. Specifically, the authors calculate the set I of IoU values between the proposed regions with the ground truth; then, they store the K-th largest IoU value into the set I. After C iterations, the new threshold value will be updated as the mean of all values in set I. As mentioned in Section

3.1, the authors also improve the regression task by adjusting β in SmoothL1 loss. Still, in this study, we focus on the IoU threshold because of two reasons:

- Our research only focuses on applying the Dynamic Label Assignment method to multi-stage (Cascade R-CNN) detection methods, thereby increasing the quality of selected anchor boxes. Better data for RPN lead to better quality proposals. Besides, when applied on Cascade R-CNN, a method with many stages, applying Dynamic Smooth L1 on one or more stages is an issue that needs further research.

- The performance when applying both two components on Cascade R-CNN in the first stage is lower than applying only Dynamic Label Assignment on the UIT-DODV dataset. Maybe adjusting many parameters on a multi-stage detector is not suitable. The experiment using both two components is shown in Table 1.

Table 1: Comparison between Dynamic Label Assignment + SmoothL1 and Dynamic Label Assignment

Method	**AP**				**AP**	**AP50**	**AP75**
	Caption	**Figure**	**Table**	**Formula**			
Dynamic Label Assignment + SmoothL1	75.1	83.8	94.1	50.7	76	90.3	82.5
Dynamic Label Assignment	75.5	85.6	94.6	54.2	77.5	91.5	84.3

Therefore, we present the IoU-based Dynamic Training algorithm based on the authors' Dynamic Training algorithm in [21], through which we remove the β update in the SmoothL1 loss function and only keep the IoU adjusting.

Regarding complexity, in each iteration, Pseudocode 1 only takes considerable time to compute the IoU output between the prediction set P and the ground truth set G, and the original research has performed this calculation function. Regarding the Cascade R-CNN, we only record the K-th highest IoU K_I among the calculated IoU values, and we update the IoU threshold after C iterations. This took no extra computation time.

3.3. Hyperparameters selection

Based on Pseudocode 1, there are vital hyperparameters in our study: IoU threshold K_I; update iterations C. Choosing too low or high K_I leads to sample imbalance. The low K_I means a high IoU threshold. In this case, the number of positive samples will be less than the negative samples. The high K_I means the low IoU threshold leads to the number of positive samples being higher. Both cases are not good for RPN. The authors in [21] choose $K_I = 64/75/100$, which means record the 64/75/100-th highest IoU corresponding to 12.5%/15%/20% of a batch of proposal boxes. The authors supposed that the 75-th highest IoU is suitable for experiments, but they did not try some smaller values. So that we extend their experiments by trying $K_I = 40, 50, 60, 70$, observe the result and choose the best for further experiments. Different C values achieves the approximately same result [21], so we keep $C = 100$.

Algorithm 1: IoU-based Dynamic Training

```
Input  : Proposal set $P$, ground-truth set $G$.
         IoU threshold top-k $K_I$, update iteration count $C$.
Output: Trained object detector $D$.
Initialize IoU threshold $T_+, T_-$. Define empty list $L_I$ for recording the IoUs.
for $i = 0$ to iterations do
    Obtain matched IoUs $I$ between $P$ and $G$
    Select thresholds $I_k$ based on the $K_I$
    Record corresponding values, add $I_k$ to $L_I$
    if $i\%C == 0$ then
        Update $T_+ = Mean(L_I)$
        Update $T_- = Mean(L_I)$
        Update $L_I = \varnothing$
    end
    Train the network with new $T_+, T_-$
end
Improved object detector $D$
```

3.4. Experiments on stages of Cascade R - CNN

As mentioned in Subsection 2.1.3, we mainly focus on applying Dynamic Label Assignment on Cascade R-CNN to achieve better performance. Since Cascade R - CNN has three stages of processing; we do experiments using Dynamic Label Assignment from one to all three stages for the comparison. In the original, the initial IoU threshold of the three stages is 0.5, 0.6, and 0.7, respectively. The Dynamic Label Assignment only adjusts the applied stages and keeps stable at other stages. Each experiment from one to three stages are trained with K_I = 40, 50, 60, 70. Results are reported in Section 4.

4. EXPERIMENTS

4.1. Datasets

4.1.1. SeaShips dataset

SeaShips dataset [17] includes 7000 images 1920 × 1080 including ore carriers, bulk carriers, general cargo ships, container ships, fishing boats, and passenger ships collected by a system of 156 surveillance cameras at 50 Various locations in Hengqin Island, Zhuhai Province, China. Three cameras are normally installed in each position, including one high-definition, low-light dome camera and two high-definition ray cameras; the remaining six are panoramic cameras. The dataset is divided at the rate of 70% - 18% - 12% corresponding to 3 sets of train - valid - test. In Table 2 we statistics the number of object envelopes in classes in the dataset.

Table 2: Our statistics of the number of boxes in the SeaShips dataset

Set \ Classes	bulk cargo carrier	container ship	fishing boat	general cargo ship	ore carrier	passenger ship
Train	1348	627	1556	1064	1534	317
Val	363	162	350	271	405	94
Test	241	112	284	170	260	63
Total	1952	901	2190	1505	2199	474

4.1.2. UIT-DODV dataset

UIT-DODV [18] is introduced by Truong Dieu et al., and it is the first Vietnamese document image dataset, including 2,394 images with four classes: Table, Figure, Caption, Formula. UIT-DODV converted 1,696 images from PDF with size 1654 × 2338, 247 images scanned from the physical scanner and expanded with 451 images scanned from the smartphone. In Table 3 we statistics the number of object envelopes in classes in the dataset.

Table 3: Our statistics of the number of boxes in the UIT-DODV data

Set \ Classes	Caption	Figure	Table	Formula
Train	2106	1144	1048	1364
Val	334	212	151	86
Test	1176	680	558	339
Total	3616	2036	1757	1789

4.1.3. MS-COCO dataset

We additionally use the MS-COCO [8] dataset version 2017 to perform experiments to demonstrate the effectiveness of the proposed method. The dataset contains 1.5 million bounding boxes labelled as 80 classes. The train/valid ratio of the dataset is 118K/5K. We evaluate our method on the MS-COCO test-dev (41K images) dataset by submitting our result on the scoring server*. MS-COCO is the standard dataset for evaluating current object detection methods.

4.2. Metrics

To evaluate object detection methods, we compute Average Precision (**AP**) for each class which is the average of AP with IoU threshold $i \in [0.5; 0.95]$. Besides, we also compute the mean of **AP** of all classes with different IoUs: **AP** is the average of (**AP**) of all classes with IoU threshold $i \in [0.5; 0.95]$; **AP@50** and **AP@75** have IoU thresholds of 0.5 and 0.75, respectively.

It can be seen that the larger the IoU value, the higher the accuracy requirement between the prediction envelope and the ground truth will be. Measuring AP and AP at thresholds of 0.5 and 0.75 will give an overview of the effectiveness of the object detection model.

*https://competitions.codalab.org/competitions/20794

4.3. Evaluation

In this section, we report our all experiments. We use ResNet-101 as the backbone architecture for all experiments, train 24 epochs on NVIDIA Tesla V100 SXM2 16GB, and other hyperparameters we set as default. For convenience, we use the MMDetection toolbox [3], an open-source which is a part of the OpenMMLab project built by Pytorch 1.3+.

Table 4: Evaluation on multiple stages on Cascade R-CNN with values of K_I are 40, 50, 60, 70 respectively. The highest result also emphasized by corresponding colors: **AP**, **AP@50**, **AP@75.**

Stage	K_I	**40**	**50**	**60**	**70**
1 Stage	**AP**	78.7	**79.2**	78.9	78.8
	AP@50	97.9	98.1	98.1	98.2
	AP@75	91	**92.1**	91.7	91.3
2 Stages	**AP**	78.9	79	78.8	79
	AP@50	97.9	**98.3**	97.9	98.1
	AP@75	91	91.4	92.1	91.3
3 Stages	**AP**	78.6	78.7	78.4	79
	AP@50	97.9	98.2	98.1	98.2
	AP@75	91	91.4	92.1	91.3

Table 5: Evaluation on Faster R - CNN (Faster), Cascade R - CNN (Cascade) and applying Dynamic Label Assignment with $K_I = 50$. We also emphasize the highest AP of six classes and **AP**, **AP@50**, **AP@75** by corresponding color: ore carrier, bulk cargo carrier, general cargo ship, container ship, fishing boat, passenger ship, **AP**, **AP@50**, **AP@75.**

Method	**AP per class**						**AP**	**AP@50**	**AP@75**
	ore carrier	**bulk cargo carrier**	**general cargo ship**	**container ship**	**fishing boat**	**passenger ship**			
Faster R-CNN	74.5	79.2	78.5	81.1	73.5	74.9	76.9	97.8	91
DLA Faster R-CNN	74.6	78.1	78.1	81.5	72.9	73	76.4	98.1	89.9
Cascade R-CNN	**79.3**	81.3	80.7	81.2	75	**76.1**	79	**98.8**	90.8
DLAFS Cascade R-CNN (ours)	78.7	**81.9**	**81.2**	**82.5**	**75.3**	75.6	**79.2**	98.1	**92.1**

Table 6: Evaluation of UIT-DODV dataset in the study of Truong Dieu et al. and compare with our method. The highest result of six classes and **AP**, **AP@50**, **AP@75** are emphasized by corresponding colors: **caption**, **figure**, **table**, **formula**, **AP**, **AP@50**, **AP@75**

Method	**AP per class**				**AP**	**AP@50**	**AP@75**
	caption	**figure**	**table**	**formula**			
Faster R - CNN[18]	57.7	79.7	91.6	45.6	68.7	86.2	76.2
CascadeTabNet[18]	73.3	83	94.3	47.5	71.8	89.1	81.6
DLA Faster R-CNN	71.5	80.7	91.2	49	73.1	90.9	80.2
DLAFS Cascade R-CNN (Ours)	**75.5**	**85.6**	**94.6**	**54.2**	**77.5**	**91.5**	**84.3**

Table 7: Evaluation on MS-COCO test-dev dataset and compare with original Dynamic R-CNN. 2× means the training schedule 2 times lengthen. The original schedule's authors are 90000 iterations, and batch size is 16 images, 2× training schedule is 180000 iterations.

Method	**Backbone**	**AP**	**AP@50**	**AP@75**	**AP$_s$**	**AP$_m$**	**AP$_l$**
Faster R-CNN + 2× [21]	ResNet-101	39.3	60.6	43.5	21.4	42.4	52.1
Dynamic R-CNN + 2× [21]		42.0	60.7	45.9	22.7	44.3	54.3
DLAFS Cascade R-CNN (ours)		**44.8**	**63.5**	**48.8**	**25.6**	**47.6**	**57.5**

4.4. Discussion

In Table 4, the Dynamic Label Assignment is applied in the first stage with $K_I = 50$ which achieves the highest **AP** and **AP@75**, **79.2%** and **92.1%** respectively. However, the highest **AP@50** is recorded at two first stages whose value is **98.3%**. It seems the proposals produced in the first stage by adjusting the IoU threshold are good enough to qualify for the higher threshold IoU, and we should not apply Dynamic Label Assignment in the next stages. $K_I = 50$ achieves the highest results in all experiments from one to all three stages, so the 50-th largest IoU threshold may reflect well the distribution of all proposals; this is a good basis for updating the new IoU threshold. From this experiment, we call our method DLAFS Cascade R-CNN (Dynamic Label Assignment on the First Stage).

In Table 5, we report the performance of DLA Faster R-CNN with $K_I = 50$ and initial IoU $= 0.5$. Can be seen that **AP** is not higher than normal training (**-0.5% AP**). But the DLAFS Cascade R-CNN performs higher results when compared with normal training(**+0.2% AP**). The **AP** metric is higher at four classes (except **ore carrier** and **passenger ship**), this method also achieve the highest **AP** and **AP@75**).

We also extend our experiments by evaluating our method on the UIT-DODV dataset introduced in the study [18]. We train and evaluate on the same train set and test set with the authors; the results are shown in Table 6. Truong Dieu et al. run experiments on several object detectors, including Faster R-CNN and CascadeTabNet, with different loss functions for the classification task. But we only focus on their evaluation using the Cross-Entropy loss function on Faster R-CNN and CascadeTabNet and compare it with our result to prove that our method also operates well on other domains. The results witness the detectors applying Dynamic Label Assignment outperforming experiments with normal training on the UIT-DODV dataset. (DLAFS Cascade R-CNN **+5.7% AP** and DLA Faster R-CNN **+4.4% AP**).

We also do experiment on the MS-COCO test-dev dataset and compare with Faster R-CNN 2× and Dynamic R-CNN 2× in [21]. The result in Table 7 shows that our DLAFS Cascade R-CNN outperform the original Dynamic R-CNN (**+2.8% AP**). Our method also achieves a result higher than Faster R-CNN 2× (**+5.5% AP**).

Some good and bad cases of our predictions on SeaShips and UIT-DODV dataset are shown in Figure 3, 4, 5, 6. Some visualization results of our predictions on MS-COCO dataset are shown in Figure 7.

5. CONCLUSION

In this study, we propose a simple but efficient DLAFS Cascade R-CNN that applies Dynamic Label Assignment on the first stage of Cascade R-CNN with $K_I = 50$. We do experiments on three datasets: SeaShips, UIT-DODV and MS-COCO. In the SeaShips dataset, we prove that our DLAFS Cascade R-CNN gives a little better performance when compared with normal Cascade R-CNN (**+0.2% AP**). In the UIT-DODV dataset, our method performs better results on both Faster R - CNN and Cascade R - CNN compared to the original study [18] (**+5.7% AP**). We also compare our results on MS-COCO with the study [21] and outperform the authors in [21] (**+2.8% AP**).

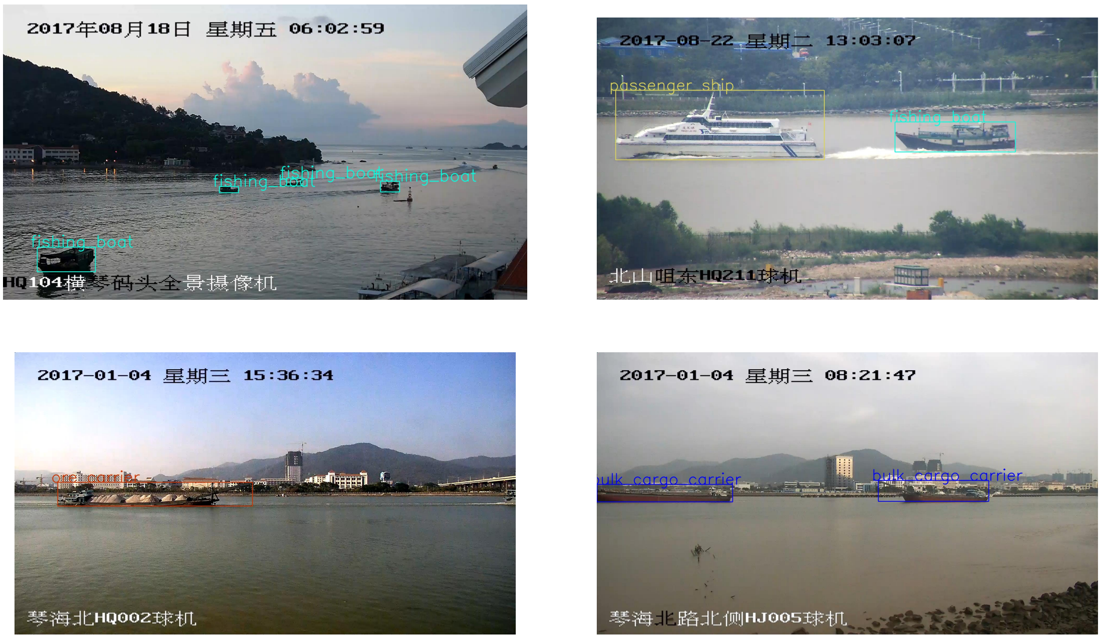
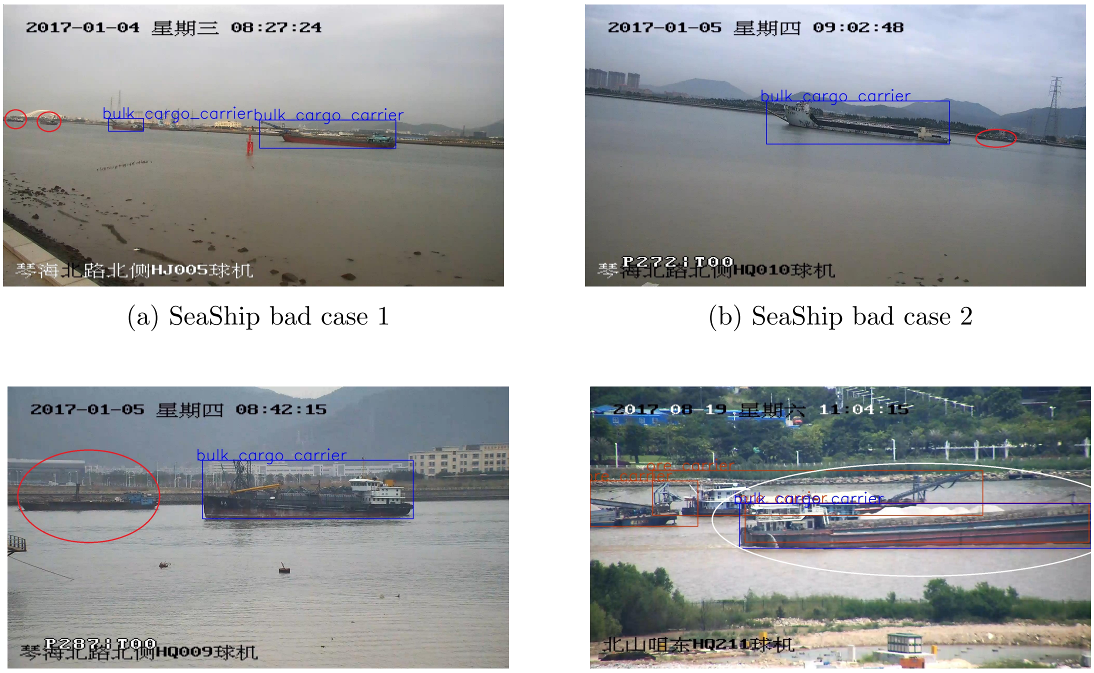


Figure 3: Some perfect cases of DLAFS Cascade R-CNN on SeaShips dataset



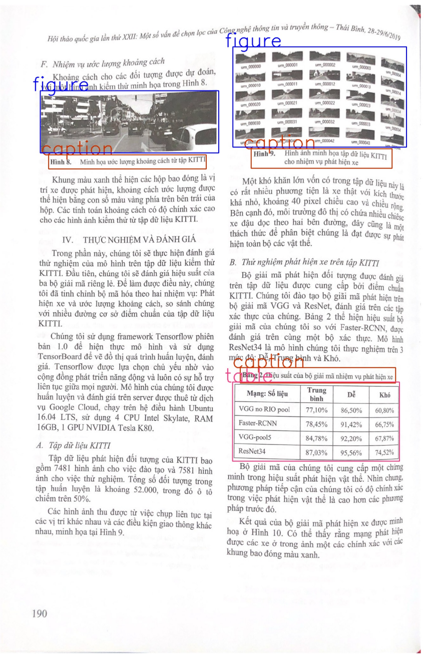
(a) SeaShip bad case 1

(b) SeaShip bad case 2

(c) SeaShip bad case 3

(d) SeaShip bad case 2

Figure 4: Some bad cases of DLAFS Cascade R-CNN on SeaShips dataset. Ignored objects and wrong objects are circled in red and white respectively.



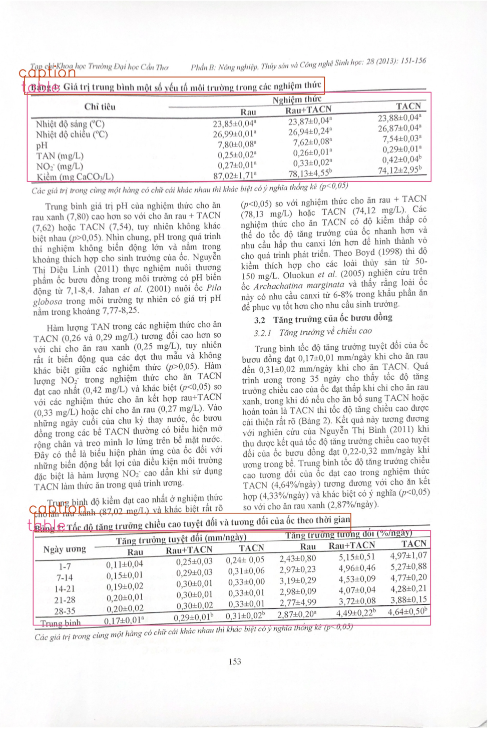
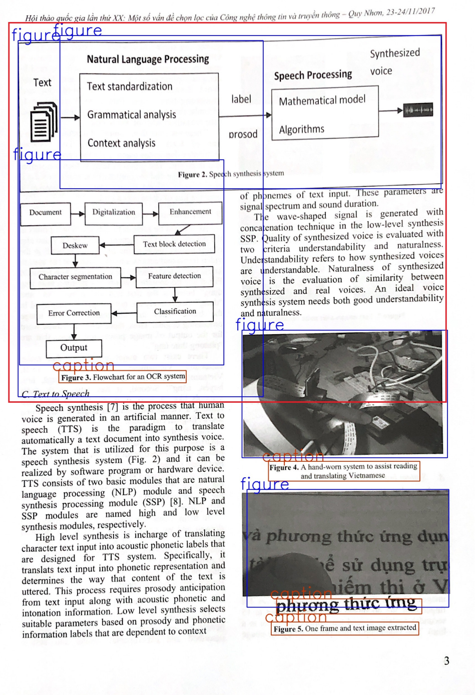
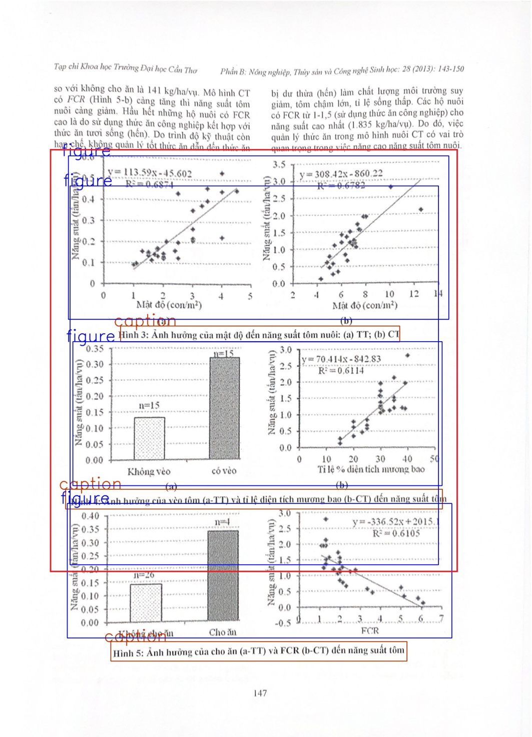


Figure 5: Some perfect cases of DLAFS Cascade R-CNN on DODV dataset



(a) DODV bad case 1



(b) DODV bad case 2

Figure 6: Some bad cases of DLAFS Cascade R-CNN on DODV dataset. Wrong predictions are zoned in red color.

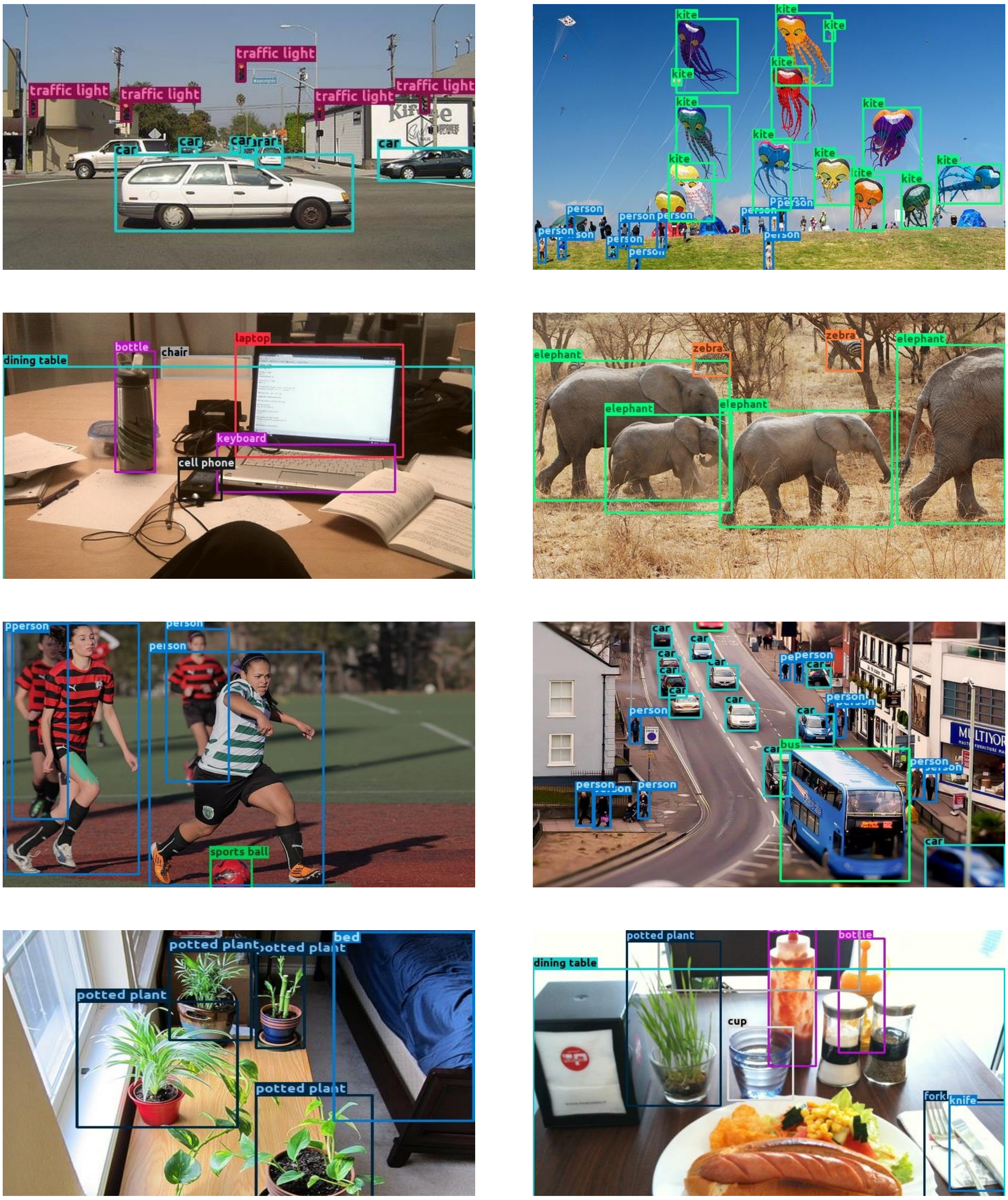


Figure 7: Visualization results of DLAFS Cascade R-CNN on MS-COCO

ACKNOWLEDGMENT

This work was supported by the Multimedia Processing Lab (MMLab) at the University of Information Technology, VNUHCM.

REFERENCES

[1] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “Yolov4: Optimal speed and accuracy of object detection,” *arXiv preprint arXiv:2004.10934*, 2020.

[2] Z. Cai and N. Vasconcelos, “Cascade r-cnn: high quality object detection and instance segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[3] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, “MMDetection: Open mmlab detection toolbox and benchmark,” *arXiv preprint arXiv:1906.07155*, 2019.

[4] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, “Centernet: Keypoint triplets for object detection,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6569–6578.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[6] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2961–2969.

[7] H. Law and J. Deng, “Cornernet: Detecting objects as paired keypoints,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 734–750.

[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *European Conference on Computer Vision.* Springer, 2014, pp. 740–755.

[9] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in *European Conference on Computer Vision.* Springer, 2016, pp. 21–37.

[10] C. C. Nguyen, G. S. Tran, T. P. Nghiem, J.-C. Burie, and C. M. Luong, “Real-time smile detection using deep learning,” *Journal of Computer Science and Cybernetics*, vol. 35, no. 2, pp. 135–145, 2019.

[11] K. Nguyen, N. T. Huynh, P. C. Nguyen, K.-D. Nguyen, N. D. Vo, and T. V. Nguyen, “Detecting objects from space: An evaluation of deep-learning modern approaches,” *Electronics*, vol. 9, no. 4, p. 583, 2020.

[12] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, “Libra r-cnn: Towards balanced learning for object detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 821–830.

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[14] J. Redmon and A. Farhadi, “Yolo9000: better, faster, stronger,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7263–7271.

[15] ——, “Yolov3: An incremental improvement,” *arXiv preprint arXiv:1804.02767*, 2018.

[16] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *arXiv preprint arXiv:1506.01497*, 2015.

[17] Z. Shao, W. Wu, Z. Wang, W. Du, and C. Li, “Seaships: A large-scale precisely annotated dataset for ship detection,” *IEEE Transactions on Multimedia*, vol. 20, no. 10, pp. 2593–2604, 2018.

[18] L. Truong Dieu, T. T. Nguyen, N. D. Vo, T. V. Nguyen, and K. Nguyen, “Parsing digitized vietnamese paper documents,” in *International Conference on Computer Analysis of Images and Patterns*, 2021.

[19] N. D. Vo, K. Nguyen, T. V. Nguyen, and K. Nguyen, “Ensemble of deep object detectors for page object detection,” in *Proceedings of the 12th International Conference on Ubiquitous Information Management and Communication*, 2018, pp. 1–6.

[20] J. Wang, K. Chen, S. Yang, C. C. Loy, and D. Lin, “Region proposal by guided anchoring,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2965–2974.

[21] H. Zhang, H. Chang, B. Ma, N. Wang, and X. Chen, “Dynamic r-cnn: Towards high quality object detection via dynamic training,” in *European Conference on Computer Vision*. Springer, 2020, pp. 260–275.

[22] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, “Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9759–9768.

Received on July 04, 2021
Accepted on May 24, 2022