CASCADE REGION PROPOSAL NETWORKS FOR OBJECT DETECTION IN THE WILD

DongMing Yang [1,2], *YueXian Zou* [1,2*]

[1] ADSPLAB, School of ECE, Peking University, Shenzhen, 518055, China
[2] Peng Cheng Laboratory, Shenzhen, 518055, China
*Corresponding author: zouyx@pkusz.edu.cn

ABSTRACT

Although significant progresses have been made in object detection on common benchmarks (i.e., Pascal VOC), object detection in the wild is still challenging due to the serious data inadequacy and imbalance. To address this challenge, we construct a cascade framework which consists of multiple region proposal networks, referred to as C-RPNs. The essence of C-RPNs is adopting multiple stages to mine hard samples and learn better classifiers. Meanwhile, a feature chain and a score chain are proposed to help learning more discriminative representations for proposals. Moreover, a loss function of cascade stages is designed to train cascade classifiers through backpropagation. Our newly proposed object detection method is evaluated on Pascal VOC and a challenging dataset of littoral birds named BSBDV 2017. Our method outperforms baseline by an obvious margin, validating its efficacy for detection in the wild.

Index Terms— object detection, cascade, hard samples mining, region proposal network, wild scenes

1. INTRODUCTION

Object detection is the most fundamental step in visual understanding. It aims at identifying and localizing objects of certain categories in images. Most of object detection approaches are trained and tested on common object detection benchmarks, i.e., PASCAL VOC [1] and MS COCO [2]. These benchmarks typically assume that objects in images are with good visibility and abundance. Obviously, this assumption is usually not satisfied in wild scenes.

Taking littoral bird images from common benchmarks and wild scenes as examples, the former are usually collected with better visibility, while the latter are usually collected via monitoring cameras with different background and camera distance. Also, the littoral birds from wild images might be in a smaller scale with more specific appearances. Moreover, different illumination and weather conditions may appear in wild scenes. For more intuitive observation, several examples focusing on littoral birds are illustrated in Fig. 1. Samples from BSBDV 2017 [3] show birds in the wild, while samples from PASCAL VOC [1] show birds in common benchmarks. The image from BSBDV 2017 is with resolution of 4912*3264, in which the heights of birds vary from 80 to 300 pixels. Images from PASCAL VOC 2007&2012 are with average resolution of 400*400, where the heights of birds are from 150 to 480 pixels. For detection approaches, such a distribution mismatch from common benchmarks to practice scenes have been observed to lead to a significant performance degradation.

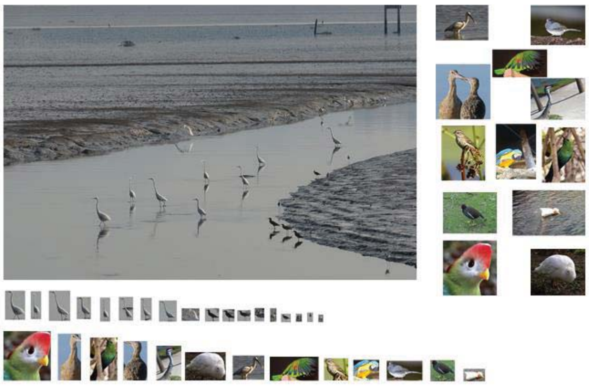


Fig. 1. Sample images: 1) one littoral bird image from BSBDV 2017 (left upper); 2) 12 images from Pascal VOC (right upper); 3) The bird objects drawn from these images (bottom). It is clearly the scales and abundances of birds are mismatched.

Although enriching training data could possibly alleviate the performance reduction, it is not favored since annotating data is expensive. To figure out the crucial elements of performance degradation in wild object detection, plenty of experiments have been conducted. We list the conclusions as follows:

1) As mentioned above, because of the smaller size, poor shooting conditions and poor abundance of objects in wild scenes, classifiers in detection approaches are unable to learn discriminative features from ground truth.

2) In a wild image, the number of negative samples (also called background samples) is much larger than that of positive samples (As shown in Fig. 1), and most of them are easy samples. Easy samples do not contribute useful learning information during training while hard samples benefit the convergence and detection accuracy. Thus, the overwhelming number of easy samples during training leads to moronic classifiers and degenerate models.

In this work, we aim to improve the precision of object detection in the wild. Based on observations above, we propose a cascade framework consists of multiple region proposal networks, referred

This paper was partially supported by Shenzhen Science & Technology Fundamental Research Programs. (No: JCYJ20160330095814461), Shenzhen Key Laboratory for Intelligent Multimedia and Virtual Reality (ZDSYS201703031405467). Special acknowledgements are given to Peng Cheng Laboratory for its support.

978-1-5386-9552-4/19/$31.00 ©2019 IEEE
DOI 10.1109/ICME.2019.00300

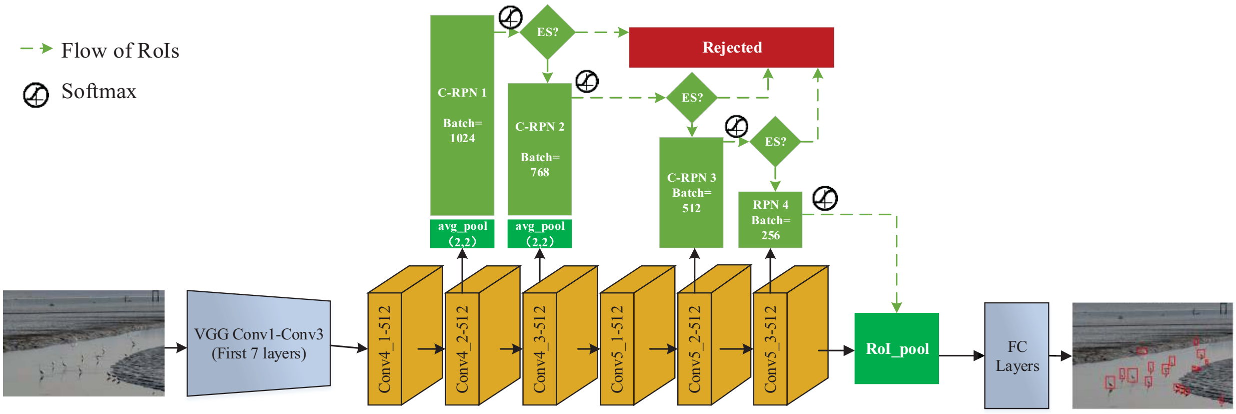


Fig. 2. An overall of our proposed C-RPNs model. We adopt VGG16 as backbone network. ES means easy samples.

to as C-RPNs. While training and testing, C-RPNs are adopted to mine hard samples and learn stronger classifiers. Multi-stage classifiers at early stages discard most of easy samples so that classifiers at latter stages focus on handling harder samples. Also, we design a feature chain and a score chain to generate more discriminative representations for proposals. Finally, a loss function of cascade stages is built to jointly learn cascade classifiers.

The contributions of this work are summarized as follows:

- A cascade region proposal networks for object detection in the wild was proposed, referred to as C-RPNs.
- A feature chain and a score chain for C-RPNs were designed to further improve classification capacity of multi-stage classifiers.
- A loss function of multiple stages was constructed to jointly learn cascade classifiers.
- Integrating the proposed components into the Faster R-CNN model, our resulting model can be trained end-to-end.

Extensive experiments have been conducted on two benchmarks, including PASCAL VOC [1] and BSBDV 2017 [3]. Our approach have provided 3.2% mAP and 11% AP gain compared with the Faster R-CNN baseline on these two benchmarks respectively. The experimental results demonstrate the effectiveness of our proposed approach for object detection in the wild.

2. RELATED WORK

2.1. Related work on object detection

We all have witnessed tremendous progresses in object detection using convolutional neural networks (CNNs) in recent years. Many CNN-based approaches have been proposed to improve performance [4-10]. Region-based CNN [4-6] approaches are referred as two-stage detectors, which have received great attention due to their effectiveness. R-CNN [4] is constrained by a selected search region. To reduce the computational complexity of R-CNN, Fast R-CNN [5] shared the convolutional feature maps among region of interest (RoI) and accelerated spatial pyramid pooling using RoI pooling layer. Renetal [6] introduced Region Proposal Network (RPN) to generate high-quality region proposals and then merged them with Fast R-CNN into a single network, referred to as Faster R-CNN. Besides, for faster detection, one-stage detectors such as YOLO [10] and SSD [9] are proposed to accomplish detection mission without region proposals.

Research shows that Faster R-CNN achieves a big success in object detection and laid the foundation for many follow-up works [8, 11-13]. For example, feature pyramid and fusion operations are adopted [11, 14] to enhance precision of detection. Deeper [15-17] or wider [18, 19] networks also benefit the detection performance. Deformable CNN [13] and Receptive Field Block Net [20] enhance the convolutional features using deformable convolutional operation and Receptive Field Block respectively. In addition, there also exists works heading in other considerations to improve the performance of object detection. For instance, large batch size [21] provides improvement in detection. SIN [22] jointly uses scene context and object relationships for promoting detection. Recently, to address the imbalanced training samples, OHEM [23] introduced an online hard example mining method for CNN based detector. From another perspective, focal loss [24] has been proposed to address the extreme foreground-background class imbalance problem in object detection, achieving the state-of-the-art.

Although excellent performance has been achieved on several benchmark datasets, such as PASCAL VOC [1] and MS COCO [2], object detection in the real world still suffers from poor precision. Works mentioned above mostly focused on the conventional setting and rarely considered the adaptation issues such as data inadequacy and imbalance for object detection in the wild.

2.2. Related work on cascade CNN

Cascade is a widely used technique to discard easy samples at early stages for learning better classification model. It is noted that, before CNNs, hand-crafted features and SVM played the most critical role in object detection. Cascade structure has been applied to SVM [25] and boosted classifiers [26, 27]. Multi-stage classifiers have been proved to be effective in generic object detection [25] and face detection [27, 28], although these multiple classifiers are not trained jointly. It has been showed that CNNs with cascade structure perform effectively on region proposal and classification [29-31] as well, in which multiple but separate CNNs were trained. After that, Qin [32] proposed a method to jointly train a cascade CNNs.

Analyzing of previous works shows that they either cannot be aggregated in the state-of-the-art convolutional framework or focus on a specific task such as face detection. Thus, confronting with the data imbalance and inadequacy problems in the wild, the research outcomes are very limited. In this work, we propose C-RPNs to mine hard samples and more discriminative features for wild object detection. Integrating with Faster R-CNN, our proposed method, to the best of our knowledge, is the first cascade model of region proposal networks for object detection in the wild.

3. PROPOSED METHOD

3.1. Overview of C-RPNs

In this study, Faster R-CNN has been adopted for our proposed C-RPNs. Faster R-CNN consists of a shared backbone convolutional network, a region proposal network (RPN) and a final classifier based on region-of-interest (RoI). For performance comparison fairness, VGG16 is taken as the backbone network [15]. Fig. 2 shows an overview of our proposed C-RPNs model. Several shared bottom convolutional layers are used for extracting convolutional features from the image (Conv1-Conv4_1). Then, C-RPNs are adopted upon four different convolutional layers, which are Conv4_2, Con4_3, Conv5_2 and Conv5_3. Since feature maps from Conv5 have the same channels but half size compared with those from Conv4, we employ an average pooling with size of 2*2 upon Conv4_2 and Conv4_3 to obtain feature maps of same resolutions for these four stages.

At stage 1, the feature map extracted from Conv4_2 are used for generating region proposals and obtaining binary classification score by a softmax function. This binary classification score estimates a sample's probabilities belonging to background or objects. Part of easy samples will be rejected at this stage. At stage 2, if a proposal has not been rejected at the former stage, then the feature map from Conv4_3 for this proposal is used for further binary classification. Similar processes are applied at stage 3 and stage 4. It is worth to point out that the stage 4 is similar to RPN from Faster R-CNN, which achieves not only binary classification but also bounding box regression. After these four stages, the proposals which have not been rejected are sent to RoI pooling layer for final detection. In this study, we set batch of each stage as 1024, 768, 512 and 256 respectively so that the stage 4 has the same batch size with RPN from Faster R-CNN.

From Fig. 2, it can be seen that C-RPNs takes different convolutional features stage-by-stage which enable it obtains different semantic information and receptive field. It is also noted that, in C-RPNs, the classifiers at shallow stages handle easier samples so that the classifiers at deeper stages focus on handling more difficult samples. The easy samples rejected by a classifier from shallow stage will not participate in the latter stages. With this design, abundant samples can be used while only hard samples been mined will go for final classification and bounding box regression, which benefits to alleviate the data imbalance problem.

To further enhance the classification capacity, a feature chain and a score chain are designed in C-RPNs, which are detailed in Section 3.2. In the end, the multi-stage classifications and bounding box regressions are learned in an end-to-end manner via a joint loss function, details are given in Section 3.3.

3.2. Feature Chain and Score Chain

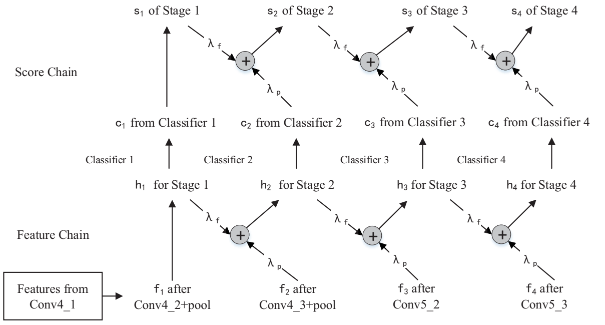


Fig. 3. The proposed feature chain and score chain of C-RPNs (VGG16 is taken as the backbone network).

Literature studies show that FPN [11] and DSSD [14] are effective for object detection using multiple convolutional layers. In this study, in order to capture the variation of features from different layers, a feature chain and a score chain at cascade stages are designed which are able to make use of features at previous stages as the prior knowledge for the features at current stage.

The implementation of feature chain and score chain is shown in Fig. 3. Let's define the number of stages as T and t is the stage index. At stage t, we denote the features from convolutional layer as f_t while features for classification as h_t. The feature chain is formulated as following:

$$h_t = \begin{cases} f_t, & t = 1 \\ \lambda_f * h_{t-1} \oplus \lambda_p * f_t, & t > 1 \end{cases} \quad (1)$$

where $\odot$ and $\oplus$ denotes the Hadamard product and summarized point to point, respectively. λ_f and λ_p are hyper parameters controlling the weight of features from former stage and present convolutional layer to generate fusional features for classification. Considering features from present convolutional layer are more helpful for classification, we set λ_f as 0.1 and λ_p as 0.9. The fused features h_t are then used for classification.

At stage t, for each proposal not been rejected at the $t - 1$ stage, we denote the score from classifier t as c_t while the output score of this stage as s_t. The designed score chain has the following formulation.

$$s_t = \begin{cases} c_t, & t = 1 \\ \lambda_f * s_{t-1} + \lambda_p * c_t, & t > 1 \end{cases} \quad (2)$$

In this implementation, it is clear to see that features and scores at current stage make use of those from previous stages which enhance the capacity of the classifiers at current stage.

3.3. Cascade Loss Function

In Faster R-CNN, training loss is composed of loss of RPN and Fast R-CNN. The former contains a binary classification loss and a regression loss while the latter contains a multi-class classification loss and a regression loss. In our method, C-RPNs adopts multi-task loss of classification and bounding-box regression to jointly optimize the detection. C-RPNs contains four binary classification loss and a regression loss. In C-RPNs, the cascade classifiers achieve to assign a sample's probabilities to background and objects. k = {0,1} is denoted to express these two class respectively. At stages t ∈ {1,2,3,4}, the set of class scores for a sample is denoted by $s = \{s_t | t = 1, \ldots, T\}$. $s_t = \{s_{t,0}, s_{t,1}\}$ are scores at stage t for background and objects respectively. Another layer at stage 4 outputs bounding box regression offsets $l = \{l^k | k = 1\}, l^k = (l_x^k, l_y^k, l_w^k, l_h^k)$ for objects. Our proposed loss function of C-RPNs has the following formulation:

$$L(s, k^*, l, l^*) = L_{cls}(s, k^*) + L_{loc}(l, l^*, k^*) \quad (3)$$

$$L_{cls}(s, k^*) = -\sum_{t=1}^{T} \alpha_t \mu_t \log(s_{t,k^*}) \quad (4)$$

where $L_{cls}(*)$ is the loss for classification and L_{loc} is the loss for bounding box regression. For L_{loc}, we use the smoothed L_1 loss [5]. For $L_{cls}(*)$, α_t is a parameter that controls the weight of loss from cascade classifiers and μ_t evaluates whether the sample is rejected in previous stages.

$$\alpha_t = \frac{\alpha_T}{10^{T-t}} \quad (5)$$

$$\mu_t = \prod_{i=1}^{t-1} [s_{t,k^*} < r] \qquad when\ t > 1, \mu_1 = 1 \quad (6)$$

Here, we set α_T=1, where $T = 4$ in C-RPNs. Since scores from deeper classifiers are more crucial for final classification than those from shallow classifiers, α_t from deeper classifiers has been distributed more weight. For μ_t, we set the r as a threshold value at each stage. $[s_{t,k^*} < r]$ will output 1 if it is true or output 0 if it is false. If a sample is rejected in previous stages, it is no longer used for training the classifier at current stage. We set r as 0.99. If $\alpha_t = \mu_t = 1$ and T = 1, then $L_{cls}(*)$ is a normal cross entropy loss. Since there is no constrain that the rejected samples must be background, few easy positive samples might also be rejected at early stages during training.

For the object detection with the proposed model, the final training loss is designed to compose the loss of C-RPNs and the loss of Fast R-CNN:

$$L_{detection} = L_{C-RPNs} + L_{roi} \quad (7)$$

where L_{C-RPNs} and L_{roi} both are composed of classification loss and regression loss. The former contains four cascade binary classification loss while the latter contains a multi-class classification loss. With this loss function, multiple classifiers and bounding box regressions are learned jointly through backpropagation.

4. EXPERIMENTS AND EVALUATIONS

4.1. Experimental setup

4.1.1. Datasets and Evaluation Metrics.

We evaluate our approach on two public object detection datasets, including PASCAL VOC 2007 [1] and BSBDV 2017 [3]. For evaluation, we use the standard mean average precision (mAP) scores with IoU thresholds at 0.5. Pascal VOC involves 20 categories. VOC 2007 dataset consists of about 5k trainval images and 5k test images, while VOC 2012 dataset includes about 11k trainval images and 11k test images. Following the protocol in [5], we perform training on the union of VOC 2007 trainval and VOC 2012 trainval. The test is conducted on VOC 2007 test set. The Birds Dataset of Shenzhen Bay in Distant View (BSBDV 2017) [3] is a great challenging dataset in the wild, consisting of 1,421 trainval images and 351 test images. BSBDV2017 contains three kinds of image resolutions, which are 2736×1824, 4288×2848 and 5472×3648 respectively. Size of birds varies greatly from 18×30 to 1274×632.

4.1.2. Implementation Details.

The Faster R-CNN is taken as our baseline, where all parameters are set according to the original publication [6] if not specified. We initialize the backbone network using a VGG16 pre-trained model on ImageNet [33] while all new layers are initialized by drawing weights from a zero-mean Gaussian distribution with standard deviation 0.01. For training on VOC 2007 test set, we use a learning rate of 0.001 for 80k iterations and 0.0001 for 30k iterations. For training on BSBDV2017, we use a learning rate of 0.001 for 50k iterations and 0.0001 for 20k iterations. We trained our model in the end-to-end manner with Stochastic Gradient Descent (SGD), where the momentum is 0.9, and the weight decay is 0.0005. Our program is implemented by Tensorflow [34] on a GPU of GeForce GTX TITAN X.

4.2. Overall performance

4.2.1. Performance on Pascal VOC benchmark.

We compare our approach with several state-of-the-art approaches in this subsection. Results in terms of mean average precision (mAP) are shown in Table 1. From Table 1, it can be seen that our model achieves the second best performance among all methods, which is 1.2% lower than that of RON [35] but 3.2% higher than that of baseline Faster R-CNN with VGG16. Besides, it is happy to see that our method outperforms ION [18] with the same backbone network which used features from Conv3_3, Conv4_3 and Conv5_3 to leverage context and multi-scale knowledge for object detection. From the table, we can see that though C-RPNs is designed aiming to improve wild detection with imbalance data, it gets competitive performance on common benchmarks.

4.2.2. Performance on BSBDV 2017.

Table 2 shows the comparisons of C-RPNs with state-of-the-arts on BSBDV 2017. From Table 2, we can see that our model performs best and its average precision (AP) is 3.4% higher than the second best (FPN [11]). More specifically, the AP of C-RPNs is 70.3%, which obtains 11% performance gain compared with that of Faster R-CNN. It is noted that our C-RPNs gets slightly lower mAP than that of RON [35] on VOC 2007, but it outperforms RON by a margin of 12.3% on BSBDV 2017. Also, C-RPNs is 8.8% and 3.4% higher than that of R-FCN [7] and FPN [11] respectively. These results demonstrate that our C-RPNs is more competitive in object detection in the wild.

4.3. Ablation studies

Table 1. Results on PASCAL VOC 2007 test set. 07+12: union of Pascal VOC07 trainval and VOC12 trainval.

Method	Trainset	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Fast R-CNN [5]	07+12	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
Faster R-CNN [6]	07+12	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
SSD500 [9]	07+12	75.1	79.8	79.5	74.5	63.4	51.9	84.9	85.6	87.2	56.6	80.1	70.0	85.4	84.9	80.9	78.2	49.0	**78.4**	72.4	84.6	75.5
ION [18]	07+12	75.6	79.2	**83.1**	**77.6**	65.6	54.9	85.4	85.1	87.0	54.4	80.6	**73.8**	85.3	82.2	**82.2**	74.4	47.1	75.8	72.7	84.2	**80.4**
RON384++ [35]	07+12	**77.6**	**86.0**	82.5	76.9	**69.1**	59.2	**86.2**	85.5	87.2	59.9	81.4	73.3	85.9	**86.8**	**82.2**	**79.6**	**52.4**	78.2	**76.0**	**86.2**	78.0
SIN [22]	07+12	76.0	77.5	80.1	75.0	67.1	62.2	83.2	86.9	**88.6**	57.7	**84.5**	70.5	**86.6**	85.6	77.7	78.3	46.6	77.6	74.7	82.3	77.1
C-RPNs (ours)	07+12	76.4	78.6	79.5	76.3	66.5	**63.2**	84.6	**87.8**	87.8	**60.2**	83.3	71.7	85.5	86.1	81.4	79.2	49.2	75.2	73.9	83.1	75.7

Table 2. Performance Comparison on BSBDV 2017.

Method	Backbone Network	AP (%)
SSD500 [9]	VGG16 reduce	42.0
Faster R-CNN [6]	VGG16	59.3
RON [35]	ResNet-101	58.0
R-FCN [7]	ResNet-50	61.5
FPN [11]	ResNet-50	**66.9**
SIN [22]	VGG16	58.4
C-RPNs （ours)	VGG16	**70.3**

Table 3. The impact of cascade stages (BSBDV 2017)

AP of C-RPNs (%)	69.5	69.9	70.3
C-RPNs with Stage 4	√	√	√
C-RPNs with Stage 3	√	√	√
C-RPNs with Stage 2		√	√
C-RPNs with Stage 1			√

Table 4. The impact of feature/score chain (BSBDV 2017).

AP of C-RPNs (%)	69.4	70.0	69.8	70.3
Feature Chain		√		√
Score Chain			√	√

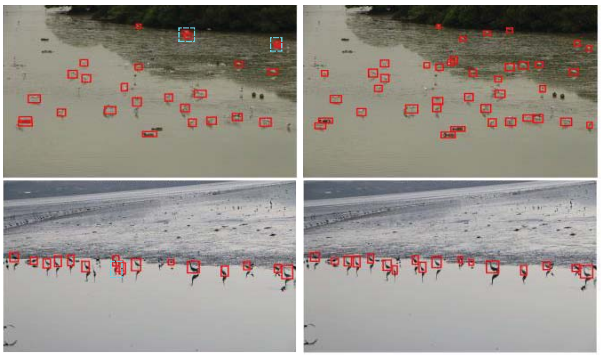
To further evaluate the individual effect of components of our C-RPNs, we analyze the object detection performance affected by the cascade stages as well as feature chain and score chain. We use BSBDV 2017 in this study.

4.3.1 Effects of cascade stages

Table 3 summarizes the performance of our C-RPNs with different number of cascade stages. With stage 3 and stage 4, C-RPNs achieves AP of 69.5% which already outperforms the baseline Faster R-CNN. Adding stage 2 and stage 1 yields AP of 69.9% and 70.3% respectively, and it brings 0.4% and 0.4% performance gain respectively. These results validate that more cascade stages and classifiers in the C-RPNs benefit object detection in the wild.

4.3.2 Effects of feature chain and score chain

Table 4 shows the performance of our C-RPNs with or without feature chain and score chain. We set the same parameters for C-RPNs with previous sections but control the usage of feature chain and score chain separately. As shown in Table 4, feature chain is found to be effective in C-RPNs, which brings 0.6% performance gain. When we adapt score chain but without feature chain, the AP is 0.4% higher, which illustrates the efficiency of using score chain as well. The adjustment boosts the performance by 0.9% while both feature chain and score chain are used.



(a) Faster R-CNNR (b) our C-RPNs

Fig. 4. Detection results of Faster R-CNN (column 1) and our proposed C-RPNs (column 2) on BSBDV 2017.

4.4. Qualitative Examples

For visualization purpose, several examples of detection results on BSBDV 2017 are given in Fig. 4. The columns from left to right are respectively expressed as the results of Faster R-CNN and C-RPNs. According to the ground truth, there are 46 and 22 birds in the top and bottom images, respectively. Compared with the results detected with Faster R-CNN, our method brings 16 and 2 more birds detected in two images respectively. Meanwhile, dotted boxes show samples are detected with more than one box, three in the left images and none in the right images. These results indicate that our method is able to generate more precise bounding boxes.

5. CONCLUSION

In this paper, we have constructed a C-RPNs, an effective approach for object detection in the wild. The essence of our C-RPNs lies in adopting cascade region proposal networks to discard easy samples and learn stronger classifiers. Moreover, a feature chain and a score chain at multiple stages are proposed to help generating more discriminative representations for proposals. Finally, a loss function of cascade stages is designed to jointly learn cascade classifiers. Extensive experiments have been conducted to evaluate our C-RPNs on a common benchmark (Pascal VOC) and a challenging dataset of littoral birds (BSBDV 2017). Our C-RPNs outperforms the Faster R-CNN baseline by an obvious margin, demonstrating its efficacy. for object detection in the wild.

6. REFERENCES

[1] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision,* vol. 88, no. 2, pp. 303-338, 2010.

[2] T. Lin *et al.*, "Microsoft COCO: Common Objects in Context," *European Conference on Computer Vision,* pp. 740-755, 2014.

[3] W. Guan, "Multi-Scale Object Detection with Feature Fusion and Region Objectness Nwtwork," *International Conference on Acoustics, Speech, and Signal Processing*, 2018, pp. 2596-2600.

[4] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," *Computer Vision and Pattern Recognition,* pp. 580-587, 2014.

[5] R. Girshick, "Fast R-CNN," *Computer Science,* 2015.

[6] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 39, no. 6, pp. 1137-1149, 2017.

[7] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object Detection via Region-based Fully Convolutional Networks," *Neural Information Processing Systems,* pp. 379-387, 2016.

[8] S. Gidaris and N. Komodakis, "Object Detection via a Multi-region and Semantic Segmentation-Aware CNN Model," *International Conference on Computer Vision,* pp. 1134-1142, 2015.

[9] W. Liu *et al.*, "SSD: Single Shot MultiBox Detector," *European Conference on Computer Vision,* pp. 21-37, 2016.

[10] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *Computer Vision and Pattern Recognition,* pp. 779-788, 2016.

[11] T. Lin, P. Dollar, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature Pyramid Networks for Object Detection," *Computer Vision and Pattern Recognition,* pp. 936-944, 2017.

[12] L. Zhang, L. Lin, X. Liang, and K. He, "Is Faster R-CNN Doing Well for Pedestrian Detection?," *European Conference on Computer Vision,* pp. 443-457, 2016.

[13] J. Dai *et* al., "Deformable Convolutional Networks," *International Conference on Computer Vision,* pp. 764-773, 2017.

[14] C. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "DSSD : Deconvolutional Single Shot Detector," *arXiv: Computer Vision and Pattern Recognition,* 2017.

[15] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *International Conference on Learning Representations,* 2015.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Computer Vision and Pattern Recognition,* pp. 770-778, 2016.

[17] C. Szegedy *et al.*, "Going deeper with convolutions," *Computer Vision and Pattern Recognition,* pp. 1-9, 2015.

[18] S. Bell, C. L. Zitnick, K. Bala, and R. B. Girshick, "Inside-Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks," *Computer Vision and Pattern Recognition,* pp. 2874-2883, 2016.

[19] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," *British Machine Vision Conference,* 2016.

[20] S. Liu, D. Huang, and Y. Wang, "Receptive Field Block Net for Accurate and Fast Object Detection," *European Conference on Computer Vision,* pp. 404-419, 2018.

[21] C. Peng *et al.*, "MegDet: A Large Mini-Batch Object Detector," *Computer Vision and Pattern Recognition,* 2018.

[22] Y. Liu, R. Wang, S. Shan, and X. Chen, "Structure Inference Net: Object Detection Using Scene-Level Context and Instance-Level Relationships," *Computer Vision and Pattern Recognition,* 2018.

[23] A. Shrivastava, A. Gupta, and R. B. Girshick, "Training Region-Based Object Detectors with Online Hard Example Mining," *Computer Vision and Pattern Recognition,* pp. 761-769, 2016.

[24] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollar, "Focal Loss for Dense Object Detection," *International Conference on Computer Vision,* pp. 2999-3007, 2017.

[25] P. F. Felzenszwalb, R. B. Girshick, and D. A. Mcallester, "Cascade object detection with deformable part models," *Computer Vision and Pattern Recognition*, 2010, pp. 2241-2248.

[26] P. Dollar, R. Appel, S. J. Belongie, and P. Perona, "Fast Feature Pyramids for Object Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 36, no. 8, pp. 1532-1545, 2014.

[27] R. Xiao, L. Zhu, and H. Zhang, "Boosting chain learning for object detection," *International Conference on Computer Vision*, 2003, pp. 709-715.

[28] L. D. Bourdev and J. Brandt, "Robust object detection via soft cascade," *Computer Vision and Pattern Recognition*, 2005, vol. 2, pp. 236-243.

[29] W. Ouyang *et al.*, "DeepID-Net: Deformable deep convolutional neural networks for object detection," *Computer Vision and Pattern Recognition,* pp. 2403-2412, 2015.

[30] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," *Computer Vision and Pattern Recognition*, 2015, pp. 5325-5334.

[31] B. Yang, J. Yan, Z. Lei, and S. Z. Li, "CRAFT Objects from Images," *Computer Vision and Pattern Recognition,* pp. 6043-6051, 2016.

[32] H. Qin, J. Yan, X. Li, and X. Hu, "Joint Training of Cascaded CNN for Face Detection," *Computer Vision and Pattern Recognition*, 2016, pp. 3456-3465.

[33] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Feifei, "ImageNet: A large-scale hierarchical image database," *Computer Vision and Pattern Recognition*, 2009, pp. 248-255.

[34] M. Abadi *et al.*, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," *arXiv: Distributed, Parallel, and Cluster Computing,* 2015.

[35] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen, "RON: Reverse Connection with Objectness Prior Networks for Object Detection," *Computer Vision and Pattern Recognition,* pp. 5244-5252, 2017.