Atrous Faster R-CNN for Small Scale Object Detection

Tongfan Guan
School of Automation. Beijing Institute of Technology
State Key Laboratory of Intelligent Control and Decision of Complex System
Beijing, China
e-mail: aeolus.guan@gmail.com

Hao Zhu
School of Automation. Beijing Institute of Technology
State Key Laboratory of Intelligent Control and Decision of Complex System
Beijing, China
e-mail: richie.zhu08@gmail.com

***Abstract*—Deep Convolutional Neural Networks based object detection has made significant progress recent years. However, detecting small scale objects is still a challenging task. This paper addresses the problem and proposes a unified deep neural network building upon the prominent Faster R-CNN framework. This paper has two main contributions. Firstly, an Atrous Region Proposal Network (ARPN) is proposed to explore object contexts at multiple scales by sliding a set of atrous filters with increasing dilation rates over the last convolutional feature map. Secondly, to enrich the representations of small scale image regions, this paper incorporates atrous convolution into Fast R-CNN and proposes a Dense Fast R-CNN (DFRCN), that improves the resolution of the ROI-pooled convolutional feature maps without increasing the number of parameters. In combination of the two, this paper proposes a unified network termed as Atrous Faster R-CNN. On PASCAL object detection challenge dataset, our method achieves superior performance to the state of the arts, especially for small scale objects.**

Keywords-component; deep learnng; object detection; small scale; atrous convolution

I. INTRODUCTION

The variations of object location, scale and appearance, together with background clutter, make it challenging to detect objects in images efficiently. Typically, the object detection task generally consists of three stages, *select candidate regions*, based on which *extract features*, and then *classify using pre-trained models*. Traditional methods usually employ the sliding window scheme to select candidate regions and then classify the manually engineered features using SVM, AdaBoost, etc. [1][2][3].

Recently, many detection methods build upon Deep Convolutional Neural Networks (DCNNs), which have significantly boosted the detection performance. One of the most prominent work is the Region-based Convolutional Neural Networks (R-CNN) [4]. R-CNN selects around 2,000 region proposals using selective search [5], and warps each of them into 227 × 227 image patch to fit neural networks. R-CNN then performs a forward pass on every single warped image patch independently. In spite of its striking success, R-CNN suffers from low efficiency for both training and deployment. In order to reduce computational load, SPP-NET [6] and Fast R-CNN (FRCN) [7] are proposed. Both of them perform a single forward pass on the entire image before fully-connected (*fc*) layers and then pool features for each candidate proposal from the last convolutional (*conv*) feature map. The pooled features are finally fed to a classifier, consisting of *fc* layers, to predict class-specific probabilities and bounding box regressions. As the result, FRCN achieves hundreds of times speedup. The candidate proposal selection process is the efficiency bottle- neck of FRCN. Then Faster R-CNN [8] is proposed using DCNN for the region proposal generation and runs at 7 fps on a Nvidia K40 GPU. Other than the R-CNN framework, some works formulate the object detection as a regression task defined on a set of pre-determined bounding boxes, such as YOLO [9] and SSD [10], and have achieved competitive performance in real time.

Although object detection has made breakthrough progress in recent years, detecting small scale objects remains an open problem. In particular, we consider two challenging issues in the detection of small scale objects. Firstly, existing object proposal mechanisms, based on low-level cues or DCNNs, suffer a low recall on small scale objects [11]. The standard approach to handle this is to present rescaled versions of the same image to the object proposal generator. But the trivial trick is usually computationally prohibitive. The second problem is the resolution loss in deep conv feature maps caused by repeated down sampling of DCNNs. High-level features in deep conv feature maps are important for classification. However, in both Fast R-CNN and Faster R-CNN, the projected regions of small objects on deep conv feature maps usually would be too small to contain enough information for a reliable classification. To deal with this issue, Yang et al. [12] propose a scale dependent pooling (SDP) technique, in which small proposals pool features from the shallow conv feature maps (with high resolution) and large proposals pool features from the deep ones (with low resolution). The pooled features from different conv layers are then classified by corresponding classifiers. This technique indeed improves the small-scale object detection precision at the cost of introducing large amount of network parameters. This issue is far from being perfectly solved.

Towards solving the problems, this paper proposes a unified deep neural network building upon Faster R-CNN with two main contributions. Firstly, we propose an Atrous Region Proposal Network (ARPN) that slides a set of atrous filters with increasing dilation rates over the last conv feature map. Atrous filters are able to enlarge receptive fields without increasing the number of parameters or operations, which enables ARPN to efficiently capture

978-1-5090-5954-6/17 $31.00 © 2017 IEEE
DOI 10.1109/ICMIP.2017.37

object contexts at multiple scales and consequently improves performance against scale variation in object proposal selection stage. Secondly, this paper proposes a Dense Fast R-CNN (DFRCN) method incorporating atrous convolution into FRCN. DFRCN has the advantage of being able to fine-tune from a pre-trained model comparing to SDP, in which parts of the classifiers start training from the scratch. We conduct extensive experiments with the proposed methods on the PASCAL VOC dataset, showing a superior detection precision to Fast R-CNN and Faster R-CNN, especially for small scale objects.

The remainder of this paper is organized as follows: We present the Atrous Region Proposal Network in Section II. The Dense Fast-RCNN is explained in detail in Section III. Our experimental evaluation and results are presented in Section IV.

TABLE I. VOC 2007 TEST AVERAGE RECALL (%) OF RPN AND ARPN, AS A FUNCTION OF BOX AREA. WE CONSIDER 2 CASES WHEN WE SELECT 2000 OR 300 CANDIDATE PROPOSALS PER IMAGE. BOX AREA: "SMALL": [0, 32^2), "MEDIUM": [32^2, 96^2), "96-128": [96^2, 128^2), "128-256": [128^2, 256^2), "256-512": [256^2, 512^2), "ALL": [0, 512^2). BETTER NUMBERS ARE BOLD-FACES.

method	#proposal	small	medium	96-128	128-256	256-512	all
RPN	2000	26.9	55.4	60.1	63.1	66.3	59.4
ARPN	2000	**33.8**	**56.8**	**63.7**	**65.8**	**66.8**	**61.5**
RPN	300	25.3	50.6	53.4	61.2	66.2	56.4
ARPN	300	**30.4**	**51.8**	**59.8**	**65.1**	**66.8**	**59.1**

TABLE II. VOC 2007 TEST DETECTION AVERAGE PRECISION (%) OF BASELINES (FRCN AND FASTER R-CNN) AND OUR MODELS (DFRCN AND ATROUS FASTER R-CNN, I.E. ARPN + DFRCN). BETTER NUMBERS ARE BOLD-FACED.

method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	persn	plant	sheep	sofa	train	tv	mAP
FRCN	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	70.0
DFRCN	**77.9**	**79.9**	69.3	**62.0**	**43.9**	**83.8**	**82.3**	**87.1**	**47.2**	**79.0**	**70.1**	84.5	**83.9**	**77.7**	**71.8**	**34.3**	**72.4**	73.8	**81.8**	**72.1**	**71.7**
Faster R-CNN	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	73.2
ARPN+DFRCN	**79.4**	**84.1**	**75.9**	**68.5**	**63.5**	**85.4**	**88.0**	**88.8**	**60.8**	**83.5**	**70.9**	**85.8**	**85.9**	**78.4**	**79.1**	**52.7**	**75.1**	72.6	**84.0**	**76.3**	**76.9**

II. ATROUS REGION PROPOSAL NETWORK

Object scale variation is a fundamental challenge in object detection domain. Image pyramid inputs [13] and similar techniques [14] have been applied to deal with this problem. These approaches indeed improve the performance to some extent, however, at the cost of abundant memory and computational resources. Faster R-CNN [8], a state of the art object detection framework, uses a region proposal network (RPN) to handle the scale variation. RPN shares conv feature maps with Fast R-CNN network [7] to avoid heavy computational load. In RPN, scale variation is tackled through the novel *anchor* boxes that serve as references at multiple scales and aspect ratios. Specially, the RPN generator slides a fixed set of 3 × 3 filters over the last shared conv feature map (conv5_3 for VGG16), mapping each 3 × 3 sliding window on conv5_3 to a lower-dimensional location encoding (512-d for VGG16). Then the location encoding volume is fed to two sibling 1 × 1 conv layers, which are used for classification (*cls*) and regression (*reg*) respectively. The *cls* and *reg* results for each cell on the location encoding volume are applied to corresponding reference anchors to generate multi-scale proposals. Each cell on the location encoding volume has a *receptive field* on the conv5_3 feature map and is exploited to predict proposals. However, the receptive field is fixed while proposals may vary over a wide scale range. It may be difficult to learn the location encoding with a fixed receptive field covering multi-scale object contexts. This contradiction has been experimentally shown its powerlessness when processing small scale objects [15].

In this paper, we investigate an *atrous region proposal network* (ARPN) approach (illustrated in the Fig. 1) to effectively and efficiently handle the scale variation in region proposal generation. ARPN models the scale variation explicitly through network architecture. Motivated by the *atrous spatial pyramid pooling* in [16], our ARPN scheme slides pyramid of filters, which have complementary receptive fields, over the conv5_3 feature map to capture object contexts at multiple scales and obtain a group of location encoding volume. These location encoding volumes have the same width, height and dimension. Rather than aggregating these locations encoding volumes into a single multi-scale encoding volume, each of them is then laid into a separate branch. In each branch, the location encoding volume is followed by two sibling 1 × 1 convolutional *cls* and *reg* layers to predict *objectness* scores and coordinate regressions with regard to the reference anchors at a specified scale. For instance, outputs of the top branch (see Fig. 1) are applied on the anchors of scale 64, and reference anchors of the bottom branch (see Fig. 1) are of scale 512. The *reg* and *cls* layers attached to each location encoding volume have their own set of parameters to learn scale-specific classification and regression models. As is the case of RPN, our ARPN is also

a kind of fully-convolutional network (FCN) [17] and thus is translation invariant up to the network's total stride [8].

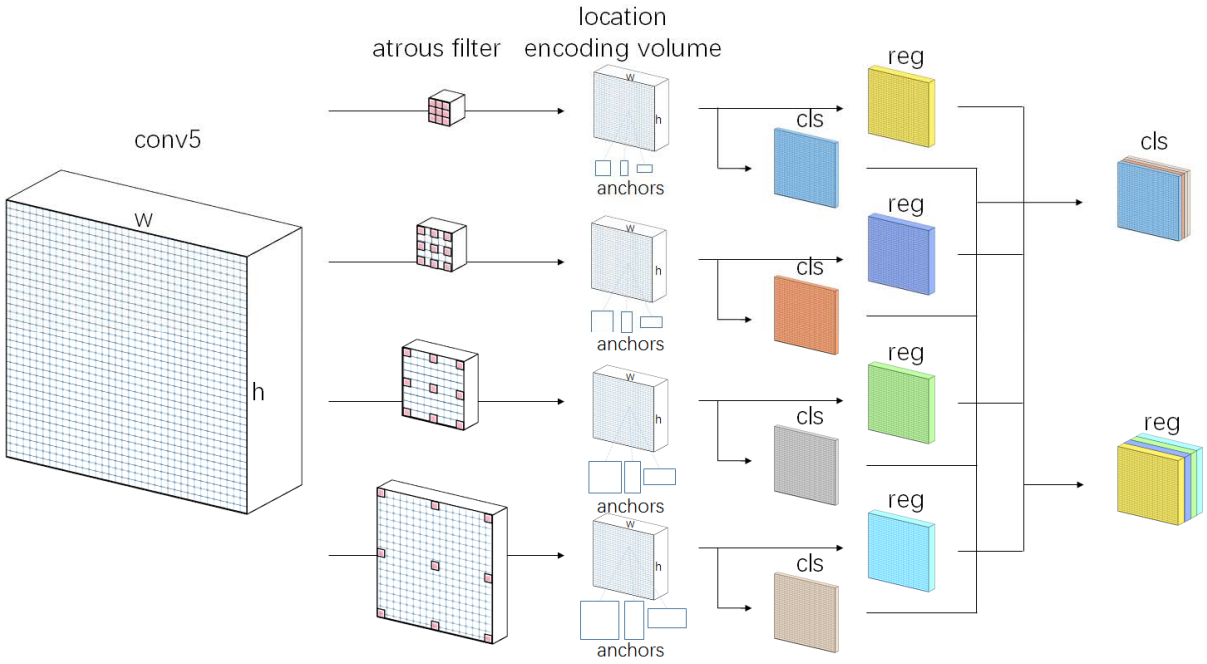

Figure 1. Filter pyramid of our Atrous Region Proposal Network (ARPN). Atrous filters are plotted with equivalent kernel size (filled with holes) and the atrous filter with larger equivalent kernel size produces the location encoding volume responsible for generating proposals of larger scale.

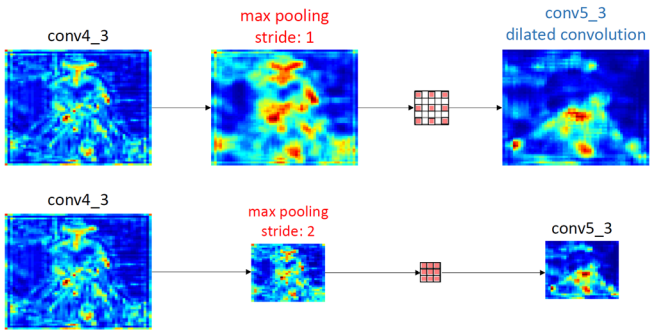

Figure 2. Convolutional feature maps of Fast R-CNN before and after incorporating atrous filters into conv5. The 3 feature maps in each row present the output of conv4_3, pool4, and conv5_3 respectively. Bottom row: original Fast R-CNN feature maps. Top row: feature maps after incorporating atrous filters into Fast R-CNN. The filters are only for illustration and do not corresponds to actual values.

The pyramid of filters we use consist of 4 filter volumes of which the receptive fields are 3×3, 5×5, 9×9 and 17×17 respectively (see Fig. 1). Instead of employing regular filters with such kernel sizes, we efficiently implement this through *atrous filters*; in other words, we advocate the use of *atrous convolution* to carry out scale-specific location encoding. Compared to regular convolution with larger filters, atrous convolution allows us to effectively enlarge the receptive fields of filters without increasing the number of network parameters or the amount of computation per position.

Atrous convolution is originally proposed for the efficient computation of the undecimated wavelet transform in the *algorithme àtrous* scheme of [18] and is recently used in the deep learning community [19] to enlarge the *receptive fields* of filters in semantic segmentation task. Taking one-dimensional signals as example, the output $y[i]$ of atrous convolution of 1-D input signal $x[i]$ with a kernel $f[k]$ of length K could be formulated as:

$$y[i] = \sum_{k=1}^{K} x[i + d * k]w[k] \quad (1)$$

The *dilation* rate, denoted as d, corresponds to the stride with which the input signal is sampled. Regular convolution can be regarded as a special case of atrous convolution when $d = 1$. The dilation algorithm is equivalent to stretching the filter by a factor of d and filling the holes with 0.

A. Training ARPN

As illustrated in Fig. 1, our ARPN model has 4 parallel branches following conv5_3. Each branch consists of an atrous convolution layer with *ReLU* activations connected to two sibling 1 × 1 convolutional *reg* and *cls* layers for predicting objectness scores and bounding box regressions, similar to [8]. The genuine kernel sizes of the atrous filters are all 3 × 3, the dilation rates are 1, 2, 4 and 8 for branches from top to bottom, corresponding to the receptive fields of 3 × 3, 5 × 5, 9 × 9 and 17 × 17 respectively. The *reg* outputs from all branches are then concatenated together along the

channel axis and the same applies to the *cls* outputs for *example sampling* in training stage.

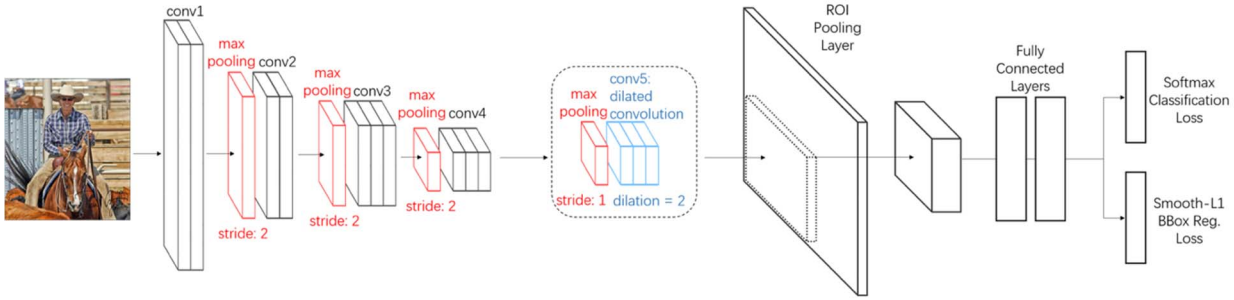

Figure 3. Dense Fast R-CNN architecture. Two transformations are performed on the basis of FRCN. Firstly, the striding parameter of pool4, the max-pooling layer that just precedes conv5, is set to 1 instread of 2. Secondly, the filters in conv5 is dilated by a factor of 2 to fit the upsampled output of conv4

The anchors at each sliding position span 4 scales (64, 128, 256, and 512) and 3 aspect ratios (1:1, 1:2, and 2:1). They are labeled as *fg* or *bg* using the mechanism of RPN. Loss function for ARPN is also the weighted sum of normalized log cross entropy for *cls* and normalized robust loss function (smooth L1) [7] for *reg* in each minibatch because of its wide adoption. All layers in each branch are initialized from scratch. The model parameters before ARPN are initialized with the Image- Net pre-trained model VGG16 [20]. During training, gradients are back-propagated to 4 branches to update scale-specific conv filters. We explicitly enforce neurons in each branch to learn for different scale of objects by providing supervision about the scales of the reference anchors. Therefore, ARPN are able to predict region proposals at multiple scales. The experiments in section IV demonstrate that ARPN has higher Average Recall [11] than RPN. The superiority is even larger for small scale objects.

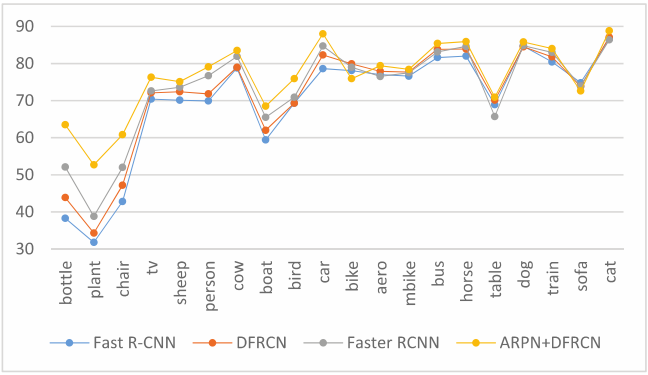

Figure 4. VOC 2007 test detection average precision (%) over all categories. The categories are arranged in ascending order measured by the *average normalized box* (see the definition in the footnote of current page)

III. Dense Fast R-CNN

DCNNs are originally designed for image classification [21] [20]. The repeated combinations of max-pooling and down- sampling in the DCNNs is essential for image classification, but results in a dramatic resolution drop of conv feature maps in deep layers that may not be able to offer enough information to detect objects, especially for small targets. The R-CNN [4] solves this problem through warping the image patch into a canonical scale (e.g. 227 × 227) at the cost of performing a forward pass on every single image patch independently. Although Fast R-CNN [7] speeds up the training and test stages compared to R-CNN and enables end-to-end training, it still does not explicitly address the reduced resolution problem.

One way to improve the performance of small-scale object detection is re-scaling. Besides this common method, some other techniques arising in dense prediction tasks, such as semantic segmentation, could be used for reference in object detection context. Long et al. [17] use VGG16 as the base network. They upsample the deep conv feature maps via a technique called *deconvolution* and then combine the upsampled deep conv feature maps with the shallow conv feature maps through skip layer fusion to get a local-to-global representation. In spite of its success, *Deconvolution Net* with skip connection is claimed to be hard to train, since it requires normalizing activations among different layers [22]. SegNet [23] develops an efficient upsample layer. The upsampled maps are sparse and convolved with trainable filters to produce dense feature maps. While the symmetric encoder-decoder architecture of the SegNet is elegant, the network depth is still doubled and the decoder sub-network can only be initialized from scratch. Atrous convolution [16] has the advantage of al- lowing computing the feature maps at any desirable resolution without increasing the network parameters. It can be applied post-hoc, once a network has been trained, but can also be seamlessly integrated with training.

This paper incorporates atrous convolution into Fast R-CNN. Fast R-CNN directly pools the features from the last conv feature map (conv5_3 for VGG16) to represent an object. The projected region on conv5_3 of each proposal is divided into a 7 × 7 spatial grid and features are pooled using max-pooling over each grid. Due to the progressive striding of max-pooling layers, a 16 × 16 image window is mapped to a single pixel at conv5_3. If an object proposal is pretty small, the same feature may repeat over several grids, which would impair the detection performance. We conjecture that *enriching the object representation would be beneficial to the performance especially for small objects*. To demonstrate our conjecture, we simply apply atrous convolution on the pre- trained VGG16 model to *upsample*

the conv5 feature maps and then fine-tune the model using detection datasets. We call our method *Dense Fast R-CNN*, which undergoes two transformations based on Fast R-CNN. The implementation details are explained as following:

- We set the stride of pool4 (max-pooling layer striding 2 just preceding conv5 layers) to 1 to avoid signal decimation. In order to ensure that the input volume and output volume of pool4 will have the same size, we set the pooling window to 3 and zero padding to 1. As illustrated in Fig. 2, the output volume of pool4 is upsampled by a factor of 2.
- The upsampled pool4 map is convolved with filters 'with holes', in which we upsample the original filters by a factor of 2 as well, and introduce zeros in between filter values. In terms of implementation details, we replace the standard convolutional layers in conv5 with atrous convolutional layers of which the dilation rate d in Equation 1 all equal to 2. The number of non-zero filter values keep constant, hence the atrous convolutional layers can also be initialized using the corresponding filters in conv5 of the pre-trained VGG16 model.

The resulted conv5_3 conv feature map (bottom row in Fig. 2) is upsampled by a factor 2, and semantically similar to the original (top row in Fig. 2) so that it makes sense to fine-tune our network initialized by pre-trained VGG16 model. Following Fast R-CNN, each proposal then pools feature from the conv5_3 map. The features are finally fed to a classifier, composed of 2 successive *fc* layers with *dropout* layers [24], for estimating class-specific probabilities and bounding box regressions. The overall model architecture is illustrated by Fig. 3.

Our method accomplishes the goal of enriching object's representation via the upsampled conv5_3 map that would be pooled for features. We simply upsample the conv5_3 map by a factor of 2 for the purpose of verifying our conjecture stated above. As expected, our conjecture is experimentally demonstrated to hold (refer Section IV for details).

IV. Experiments

We evaluate ARPN, DFRCN, and the unified Atrous Faster R-CNN that combines ARPN and DFRCN on the PASCAL VOC 2007 detection benchmark [25]. This dataset consists of about 5k trainval images and 5k test images over 20 object categories. We further argument the training set with PASCAL VOC 2012 trainval images, roughly tripling the number of images to 16.5k. For the ImageNet pre-trained network, we use the public available VGG16 model that has 13 conv layers and 3 *fc* layers. We use *Caffe* [26] for development in all experiments.

A. Performance of ARPN

Table I shows the results of RPN and our ARPN for proposal selection. Both methods start from the same pre-trained VGG16 network and use the same hyper-parameters. Following [11], average recall (AR) is used as performance metric. Since Faster R-CNN [8] didn't evaluate the performance of RPN for proposal se- lection, we train a RPN model using the code released at https://github.com/rbgirshick/py-faster-rcnn and compute ARs for it. If we generate 2000 candidate proposals for each image, ARPN achieves a higher overall AR than RPN on VOC2007 (61.5% vs 59.4%). Furthermore, ARPN improves the AR on objects of all scales thanks to ARPN branches capturing object contexts at various scales. Especially, for small scale objects (area lies in $[0, 32^2)$) that this paper concentrates on, ARPN achieves an impressive 6.9% improvement over RPN. ARPN also obtains competitive results, with an overall AR of 59.1% while selecting only 300 candidate proposals. Hence our ARPN is a better choice to support efficient object detection than RPN.

B. Performance of DFRCN

Table II shows FRCN and DFRCN detection accuracy when trained and tested using selective search proposals. We obtain FRCN results from [7] and use the same configuration to train a DFRCN model. DFRCN improves the AP on most categories and obtain the overall mAP 71.7%, which is 1.7% higher than FRCN. In addition, we observe relatively higher improvements on small objects like bottles (5.6%) and chairs (4.4%). This confirms our hypothesis that improving resolution of ROI-pooled conv feature map is beneficial to object classification in Fast R-CNN. In this paper, we use atrous convolution to increase by a factor of 2 the density of computed feature maps from a point of efficient/accuracy tradeoff. If a fast bilinear interpolation by an additional factor of 2 is followed to the last conv feature map, obtaining a high-level feature map for ROI pooling at 1/4 resolution of the original image, we suggest that detection accuracy would be improved further. This will be explored in our future work.

C. Performance of Atrous Faster R-CNN

In combination of ARPN and DFRCN, we propose a unified Atrous Faster R-CNN for object detection. Our unified network is similar to Faster R-CNN in architecture with RPN replaced by ARPN and FRCN replaced by DFRCN. Since conv5_3 is upsampled by a factor of 2 in DFRCN, we slide the pyramid filters of ARPN at the stride of 2 instead of 1 over conv5_3 for computational efficiency in our unified network. Table II shows the results of our unified network and comparisons with Faster R-CNN. We use the same configuration and learning parameters as in the previous experiments. Our approach achieves a mAP of 76.9% based on VGG16 network, which outperforms the Faster R-CNN baseline by 3.7% on average. In particular, our method significantly improves APs for small scale categories over Faster R-CNN, such as 11.4% for bottles and 13.9% for plants. We visualize the detection results in Fig. 4, in which the horizontal categories are arranged in ascending scale order from left to right measured by the *average normalized area*[1]. We observe from Fig. 4 that our

[1] Normalized area of a box is defined as the ratio of box area with the image size (width × height). The average normalized area of a category is defined as the average of normalized box among all boxes belonging to that category.

method (ARPN + DFRCN) improves the AP much more for smaller scale categories. This is a clear evidence showing the power of our Atrous Faster R-CNN on small scale objects.

V. Conclusion

In this paper, we investigate two issues that compromise the detection accuracy for small scale objects, 1) small scale objects are hard to find in candidate proposal selection stage, 2) the resolution loss of DCNNs causes that the projected regions of small scale objects on deep conv maps are usually too small to contain enough high-level feature to provide reliable classification. We incorporate two new strategies, ARPN and DFRCN, to deal with these issues. ARPN improves the recall rate for small scale objects by sliding a pyramid of filters with different receptive fields capturing object contexts at multiple scales. DFRCN upsamples the conv5_3, that is pooled for object description by each proposal, to enrich the representations of small scale objects. Our experimental evaluation clearly demonstrates the benefits of ARPN and DFRCN in small scale object detection [27]. Although considerably accurate, current deep learning based object detection methods can not fit into embedded devices. We will explore the model compression applied to object detection in the future.

References

[1] P.ViolaandM.J.Jones,"Robustreal-timefacedetection,"*International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[2] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 886–893.

[3] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Computer Vision and Pattern Recognition*, 2014.

[5] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *European Conference on Computer Vision*. Springer, 2014, pp. 346–361.

[7] R. Girshick, "Fast r-cnn," in International Conference on Computer Vision (ICCV), 2015.

[8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real- time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *arXiv preprint arXiv:1506.02640*, 2015.

[10] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed, "Ssd: Single shot multibox detector," *arXiv preprint arXiv:1512.02325*, 2015.

[11] J. Hosang, R. Benenson, P. Dolla´r, and B. Schiele, "What makes for effective detection proposals?" *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 4, pp. 814–830, 2016.

[12] F. Yang, W. Choi, and Y. Lin, "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*, 2016, pp. 2129–2137.

[13] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals for accurate object class detection," in *Advances in Neural Information Processing Systems*, 2015, pp. 424–432.

[14] G. Papandreou, I. Kokkinos, and P.-A. Savalle, "Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 390–399.

[15] Z. Cai, Q. Fan, R. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in *ECCV*, 2016.

[16] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *arXiv preprint arXiv:1606.00915*, 2016.

[17] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[18] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian, "A real-time algorithm for signal analysis with the help of the wavelet transform," in *Wavelets*. Springer, 1990, pp. 286–297.

[19] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural infor- mation processing systems*, 2012, pp. 1097–1105.

[22] W. Liu, A. Rabinovich, and A. C. Berg, "Parsenet: Looking wider to see better," *arXiv preprint arXiv:1506.04579*, 2015.

[23] V. Badrinarayanan, A. Handa, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling," *arXiv preprint arXiv:1505.07293*, 2015.

[24] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[25] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisser- man, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[26] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[27] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results," http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html.