RDSSD: Refinement Deconvolutional Single Shot Detector

Ke Guo
Northeast Normal University, Changchun, Jilin, 130117, China.
guok051@nenu.edu.cn  gk19950815@outlook.com

Abstract. The main contribution of this paper is a refinement for deconvolutional single shot detector. Our model is called refinement deconvolutional single shot detector (RDSSD). In the field of object detection, most two-stage detectors are more accurate than one-stage detectors. We use its advantages to optimize deconvolutional single shot detector. Compared with DSSD, RDSSD mainly changes two parts: the anchor fine-tuning module and the specific detection module. The anchor fine-tuning module adjusts the scale and size of the anchors. The optimized anchors are used as the input of specific detection module and accurately detect the objects. By adding these two structures, a large number of negative samples are filtered out before the specific identification, and the size of the anchors is roughly adjusted, which helps detect confusing objects. RDSSD adds the loss of the fine-tuning anchor module and the loss of the specific detection module to get the total loss. This makes our method end-to-end. We use Pascal VOC2007 and Pascal VOC 2012 datasets to train the model. Our RDSSD with 320×320 input achieves 79.3 mAP on VOC 2007.

1. Introduction
Object detection is a basic challenging task of computer vision. Since the 2012 ImageNet ILSVRC competition, professor Geoffrey Hinton of the University of Toronto led the student team, and used a convolutional neural network. They reduced the top-5 error of the ILSVRC classification task to 15.3%. Since then, using deep learning to deal with object detection task has become the mainstream. Deep learning detectors are mainly divided into two types of the one-stage approach and the two-stage approach. The one-stage approaches mainly include [2,3,4], and the two-step approaches are represented by [14].

One-stage approaches directly return the category probability of objects and bounding box offsets, just like YOLO [23] gave category and location information directly through the backbone network, without using RPN network. One-stage approaches is faster, but the accuracy is slightly lower than the two-stage approaches. The two-stage approaches take a series of candidate boxes as samples, and classify the samples through the convolutional neural network. The two-stage approaches have the advantage of higher accuracy, but the speed is not as fast as one-stage approaches. It was necessary to point out that the main reason for the difference effects between the one-stage and two-stage effects was the problem of foreground-background class imbalance [1], and people tried to solve this problem [1,6,7,9].

My method is shown in Figure 1. The anchor fine-tuning module is some convolutional layers connected to the backbone network. The anchor fine-tuning module adjusts the position and size of the anchors, and transfers the adjusted anchors to the specific detection module. The specific detection module uses the adjusted anchors as the input to predict the class and bounding box offset of the specific targets. This has obvious advantages: First predict the area where the objects may exist, and then, the objects are specifically identified based on the adjusted anchors. In this way, a large number of negative
samples are filtered out in advance, which is also the advantage of the two-stage approaches. This helps solve the foreground-background class imbalance problem. The anchor fine-tuning module is used to generate the approximate position of the objects, the specific detection module is used to specifically predict the category and location of objects. We use multi-task loss function, which is to add the category loss and bounding box loss generated by the anchor fine-tuning module and the category loss and bounding box loss generated by the specific detection module as the total loss, so RDSSD is end-to-end.

The experiments use the PASCAL VOC2007 and PASCAL VOC2012 datasets to train the model. The backbone network is resnet-101. the input image is 320×320 pixels. The test uses the PASCAL VOC2007 and the result is 79.33 mAP, relative to DSSD by 0.7 mAP. When the input dimension is 520×520 pixels, the result we get is 81.6 mAP.

2. Related Work

Today, the mainstream method for processing object detection tasks is deep learning. The object detection algorithm of deep learning is mainly divided into one-stage approach and two-stage approach.

One-stage approach directly returns the category probability and position offsets of objects. SSD[4] used multi-scale feature maps for detection and set priori boxes, which was excellent in accuracy and speed. DSSD [3] augmented SSD+Residual101 with deconvolution layers to introduce additional largescale context in object detection. It improved the detection accuracy of small objects.

The two-stage approach uses a series of candidate boxes as samples, and then classifies the samples by convolutional neural network. RCNN [10] used a selective search method [11] to generate candidate boxes, which used convolutional neural networks to extract features for each candidate box. The features were sent to the SVM classifier to determine the category possibilities, and then the positions of the candidate boxes were corrected. Spp-net [12] solved the problem that RCNN required 2000 candidate boxes to enter the convolutional neural network, and designed the spatial pyramid pooling layer. Spp-net produced fixed-length representations, regardless of the size and proportion of the input image. Spp-net was robust to object deformation. Fast R-CNN [13] designed RoI Pooling Layer and proposed a multi-task loss function. Faster R-CNN [14] proposed the RPN candidate box generation algorithm, which greatly improved the object detection speed of Faster R-CNN.
3. Network Architecture

The overall structure of RDSSD is shown in Figure 1. On the basis of DSSD, the anchor fine-tuning module and the specific detection module are mainly added. The anchor fine-tuning module is used to fine-tune the anchors. The specific detection module is used to detect objects in detail. The anchor fine-tuning module is convolutional structures connected at different levels of the backbone network. RDSSD generates fixed anchors on the selected feature maps. The fine-tuning module predicts the possibility of target existence and the offset of the bounding box based on the generated anchors, which helps to eliminate the negative anchors and roughly adjusts the position and size of the anchors. The specific detection module generates accurate category confidences and bounding box offsets based on the adjusted anchors. The two important parts of RDSSD are explained in detail below: the anchor fine-tuning module and the two-step cascaded regression.

Anchor fine-tuning module: MS-CNN [17] pointed out that improving sub-network performance can improve accuracy. Inspired by this, we tried four structures. (a) The objective function directly acts on the feature map and the L2 normalization layer. (b) the residual block with a skip connection. (c) one residual block for each prediction layer (d) two sequential residual blocks. Finally, we adopt the (c) structure. The network structure of the specific module is the same. For specific data, see Ablation Experiment.

Two-step cascaded regression: At present, most of one-stage [1, 15, 16] detection algorithms do not generate candidate boxes, and directly generate the class confidences and bounding box offsets of the objects. After a single detection, the final detection result can be obtained directly, but the accuracy is relatively low [18]. Therefore, we adopt a two-step cascade regression method: Generate n fixed anchors on k selected feature maps on the backbone network. The anchor fine-tuning module adjusts the size and position of the anchors according to the predicted 4n bounding box offsets and the probability of the existence of the targets. The anchor fine-tuning module generates n refined anchors. Taking the refined anchors as input, the specific detection module specifically predicts the category confidences and specific position bounding box offsets. In the design of the RDSSD, each feature map selected in the backbone network has the same size as the feature map connected in the deconvolution part.

4. Training and Inference

Data Augmentation: We used data augmentation methods, such as rotation, translation, zoom, random occlusion, horizontal flip, saturation, brightness, contrast, and noise perturbation. This helps to improve the robustness of the model.

Backbone Network: We use ResNet-101[19] as the backbone network of RDSSD. ResNet-101 is pretrained on the ILSVRC CLS-LOC dataset [22]. We add 4 extra residual blocks to the end of ResNet-101. The gradient descent method is used to optimize the loss function. RDSSD is also applicable to other backbone models, such as VGG [20], Inception V2 [21].

Anchor Design: To detect targets of different sizes, we preset the anchor frame on the selected feature maps of different layers. In the experiment of 320×320 input, we splice the anchor fine-tuning module on layer_3 and layer_4 and 3 extra convolutional layers of resnet-101. The corresponding feature map sizes are
40, 20, 10, 5, 3 pixels, and the preset number of anchors is 9586. In the experiment of 520×520 input, we splice the anchor fine-tuning module on layer_3 and layer_4 and 4 extra convolutional layers of resnet-101. The corresponding feature map sizes are 64, 32, 16, 8, 4, 2 pixels, and the preset number of anchors is 24560. During the training phase, we first match each ground truth to the anchor box with the best overlap score, and then match the anchor boxes to any ground truth with overlap higher than 0.5.

Hard negative mining: the anchor boxes are mostly negative anchor boxes after matching with the ground truth, which will cause foreground-background class imbalance problem. We use hard negative mining in the anchor fine-tuning module and the specific detection module. We only select hard to distinguish negative samples for training. We chose a positive-negative ratio of 1:3.

Loss function: The total loss function of RDSSD is composed of the sum of the anchor fine-tuning module loss and the specific module loss. Therefore, our final loss function is:

$$\mathcal{L}(\{p_i\}, \{x_i\}, \{c_i\}, \{t_i\}) = \frac{1}{N_a} \left( \sum_i \mathcal{L}_a(p_i, l_i \geq 1) + \sum_i \mathcal{L}_r(x_i, g_i) \right) + \frac{1}{N_s} \left( \sum_i \mathcal{L}_m(c_i, l_i) + \sum_i \mathcal{L}_r(t_i, g_i) \right)$$

where $i$ is the index of anchor; $N_a$ and $N_s$ are the number of positive anchors in the anchor fine-tuning module and the specific detection module. $g_i$ is the truth location and size of anchor. $l_i$ is the label of the real sample; $x_i$ and $p_i$ are the confidence of anchor position and size. $c_i$ and $t_i$ are the predicted class and bounding box. $L_a$ is the cross-entropy/log loss and $L_m$ is the softmax loss; $L_r$ is smooth L1 loss.

5. Experiments

5.1. Base network

We use ResNet-101 pre-trained on the ILSVRC CLS-LOC dataset. Table 1 shows which feature layers are spliced to the anchor fine-tuning module. The depth is the position of the selected layer in the network. Only the convolution and the pooling layers are considered.

| Residual-101 | conv3 | x | conv5 | x | conv6 | x | conv7 | x | conv8 | x | conv9 | x |
|--------------|-------|---|-------|---|-------|---|-------|---|-------|---|-------|---|
| Resolution320 | 40×40 | 20×20 | 10×10 | 5×5 | 3×3 | - |
| Resolution520 | 64×64 | 32×32 | 16×16 | 8×8 | 4×4 | 2×2 |
| Depth | 23 | 101 | 104 | 107 | 110 | 113 |

5.2. PASCAL VOC 2007

Our model is trained on the union of VOC 2007 trainval and VOC 2012 trainval. There are 20 classes in PASCAL VOC datasets. For 320×320 input and 512×512 input, we use a batch size of 32. In the first 80k iterations, the learning rate is 10-3, and then continue training for 40k iterations with 10-4 and 40k iterations with 10-5. Table 2 shows the results we get in the PASCAL VOC 2007 test. When the input is 320×320 pixels, the detection result obtained by RDSSD is 79.33mAP, which is an increase of 0.7 mAP compared with DSSD. When the input is 520×520 pixels, the detection result obtained by RDSSD is 81.6mAP, which is also slightly higher than DSSD.

We use pytorch1.2 to implement RDSSD. The machine uses cuda10.0 version and uses GTX1080TI graphics card. The average photo detection time is 74 milliseconds.

RDSSD has advantages in detecting objects with obvious boundaries. For example, the airplanes, chairs, trains and birds have clear boundaries.
Table 2. PASCAL VOC2007 test detection result, the input of the two-stage method in the table is 1000x600 pixels. SSD300* and SSD 512* are the latest SSD results with the new expansion data augmentation trick, which are already better than many other det.

| Method       | network | mAP | aero | bike | bird | boat | bottle | bus | car | cat | chair | cow | table | dog | home | mbike | person | plant | sheep | sofa | train | tv |
|--------------|---------|-----|------|------|------|------|--------|-----|-----|-----|-------|-----|-------|-----|------|-------|--------|-------|-------|------|-------|-----|
| Faster VGG   |         | 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 |
| Faster Residual-101 |         | 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 |
| MR-CNN VGG   |         | 76.4 | 79.8 | 80.7 | 76.2 | 68.3 | 55.9   | 85.1 | 85.3 | 89.8 | 56.7  | 87.8 | 69.4  | 88.3 | 88.9 | 80.9  | 78.4   | 41.7  | 78.6  | 79.8 | 85.3  | 72.0 |
| Faster Residual-101 | VGG    | 77.8 | 80.3 | 84.1 | 78.5 | 70.8 | 68.5   | 88.0 | 85.9 | 87.8 | 60.3  | 85.2 | 73.7  | 86.5 | 85.0 | 76.4  | 48.5   | 76.3  | 75.5  | 85.0 | 81.0  |      |
| Faster Residual-101 | Residual-101 | 80.5 | 79.9 | 87.2 | 81.5 | 72.0 | 69.8   | 86.8 | 88.5 | 89.8 | 67.0  | 88.1 | 74.5  | 89.8 | 90.6 | 79.9  | 81.2   | 53.7  | 81.8  | 81.5 | 85.9  | 79.9 |
| SSD300* VGG  |         | 77.5 | 79.5 | 83.9 | 76.0 | 69.6 | 50.5   | 85.7 | 85.1 | 88.1 | 60.3  | 81.5 | 77.0  | 86.1 | 87.5 | 83.9  | 79.4   | 52.3  | 77.9  | 79.5 | 87.6  | 76.8 |
| SSD512 Residual-101 | VGG    | 77.1 | 76.3 | 84.6 | 79.3 | 64.6 | 47.2   | 85.4 | 84.0 | 88.8 | 60.1  | 82.6 | 76.9  | 86.7 | 87.2 | 85.4  | 79.1   | 50.8  | 77.2  | 82.6 | 87.3  | 76.6 |
| DSSD321 Residual-101 | VGG    | 78.6 | 81.9 | 84.9 | 80.5 | 68.4 | 53.9   | 85.9 | 85.6 | 86.2 | 88.9  | 61.1 | 83.5  | 78.7 | 86.7 | 88.7  | 86.7   | 79.7  | 51.7  | 78.0 | 80.9  | 79.4 |
| RDSSD320 Residual-101 | VGG    | 79.3 | 79.7 | 85.2 | 77.7 | 72.9 | 58.3   | 84.6 | 86.1 | 88.4 | 64.4  | 84.7 | 76.7  | 86.9 | 86.3 | 87.2  | 81.6   | 56.6  | 83.0  | 80.2 | 86.8  | 79.4 |
| SSD512* VGG  |         | 79.5 | 84.8 | 85.1 | 81.3 | 73.0 | 57.8   | 87.8 | 88.3 | 87.4 | 63.5  | 85.4 | 73.2  | 86.2 | 86.7 | 83.9  | 82.5   | 55.6  | 81.7  | 79.0 | 86.6  | 80.0 |
| SSD513 Residual-101 | VGG    | 80.6 | 84.3 | 87.6 | 82.6 | 71.6 | 59.0   | 88.2 | 88.1 | 89.3 | 64.4  | 85.6 | 76.2  | 88.5 | 89.9 | 87.5  | 83.0   | 53.6  | 83.9 | 82.2  | 87.2  | 81.3 |
| DSSD321 Residual-101 | VGG    | 81.5 | 86.6 | 86.2 | 82.6 | 74.9 | 62.5   | 89.0 | 88.7 | 88.8 | 65.2  | 87.0 | 78.7  | 88.2 | 89.0 | 87.5  | 83.7   | 51.1  | 86.3 | 81.6  | 85.7  | 83.7 |
| RDSSD320 Residual-101 | VGG    | 81.6 | 86.6 | 87.5 | 84.9 | 75.8 | 65.5   | 87.5 | 88.2 | 89.0 | 69.8  | 85.1 | 77.6  | 88.2 | 83.3 | 84.8  | 83.7   | 55.8  | 82.9 | 81.6  | 88.1  | 79.9 |

5.3. Ablation Study on VOC2007
In order to get better results, we tried different structures, we experimented with the test results of different the fine-tuning modules: (a) the objective function directly acts on the feature map and the L2 normalization layer. (b) the residual block with a skip connection. (c) one residual block for each prediction layer (d) two sequential residual blocks. As is shown in Table 3, (c) the structure has the best results.

Table 3. Effects of various the fine-tuning module on PASCAL VOC 2007 test.

| Method                        | mAP  |
|-------------------------------|------|
| RDSSD 320 with the anchor fine-tuning module(a) | 77.6 |
| RDSSD 320 with the anchor fine-tuning module(b) | 78.6 |
| RDSSD 320 with the anchor fine-tuning module(c) | 79.3 |
| RDSSD 320 with the anchor fine-tuning module(d) | 78.7 |

5.4. Visualization
In Figure 3, we show many comparison images of DSSD320 and RDSSD320 test results. From the figure we can see that RDSSD can better detect occluded objects. RDSSD can detect these occluded objects in the background, but DSSD cannot.
6. Conclusions
In general, we propose a new object detection model called RDSSD. It adds the anchor fine-tuning module and the specific detection module on the basis of DSSD. The fine-tuning module is used to refine anchors and eliminate a large number of negative anchors. The optimized anchors are used as the input of specific detection module to accurately detect objects. RDSSD combines the advantages of one-stage approach and two-stage approach. The whole network is trained in an end-to-end approach with the multi-task loss. Finally, on the PASCAL VOC2007 and PASCAL VOC2012 datasets, the result of RDSSD320 is 79.3 mAP, and the result of RDSSD520 is 81.6 mAP.

References
[1] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection [C]//Proceedings of the IEEE international conference on computer vision. 2017:2980-2988.
[2] Redmon J, Farhadi A. YOLO9000: better, faster, stronger[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 7263-7271.
[3] Fu C Y, Liu W, Ranga A, et al. Dssd: Deconvolutional single shot detector[J]. arXiv preprint arXiv:1701.06659, 2017.
[4] Liu W, Anguelov D, Erhan D, et al. Ssd: Single shot multibox detector[C]//European conference on computer vision. Springer, Cham, 2016: 21-37.
[5] Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[C]//Advances in neural information processing systems. 2015: 91-99.
[6] Kong T, Sun F, Yao A, et al. Ron: Reverse connection with objectness prior networks for object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 5936-5944.
[7] Zhang S, Zhu X, Lei Z, et al. S3fd: Single shot scale-invariant face detector[C]//Proceedings of the IEEE international conference on computer vision. 2017: 192-201.
[8] Lin T Y, Dollár P, Girshick R, et al. Feature pyramid networks for object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 2117-2125.
[9] Zhang S, Wen L, Bian X, et al. Single-shot refinement neural network for object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 4203-4212.
[10] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2014: 580-587.

[11] Uijlings J R R, Van De Sande K E A, Gevers T, et al. Selective search for object recognition[J]. International journal of computer vision, 2013, 104(2): 154-171.

[12] Purkait P, Zhao C, Zach C. SPP-Net: Deep absolute pose regression with synthetic views[J]. arXiv preprint arXiv:1712.03452, 2017.

[13] Girshick R. Fast r-cnn[C]//Proceedings of the IEEE international conference on computer vision. 2015: 1440-1448.

[14] Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[C]//Advances in neural information processing systems. 2015: 91-99.

[15] Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 779-788.

[16] Redmon J, Farhadi A. YOLO9000: better, faster, stronger[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 7263-7271.

[17] Cai Z, Fan Q, Feris R S, et al. A unified multi-scale deep convolutional neural network for fast object detection[C]//European conference on computer vision. Springer, Cham, 2016: 354-370.

[18] Zhang S, Wen L, Bian X, et al. Single-shot refinement neural network for object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 4203-4212.

[19] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

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

[21] Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift[J]. arXiv preprint arXiv:1502.03167, 2015.

[22] Russakovsky O, Deng J, Su H, et al. Imagenet large scale visual recognition challenge[J]. International journal of computer vision, 2015, 115(3): 211-252.

[23] Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 779-788.