Expanding Receptive Field YOLO for Small Object Detection

To cite this article: Zexing Du et al 2019 J. Phys.: Conf. Ser. 1314 012202

View the article online for updates and enhancements.
Expanding Receptive Field YOLO for Small Object Detection

Zexing Du*, Jinyong Yin and Jian Yang
Jiangsu Automation Research Institute, Lianyungang, JiangSu, 222002, China
*Corresponding author’s e-mail: duzexing@outlook.com

Abstract. State-of-art object detection networks like YOLO, SSD and Faster R-CNN all have achieved great success in object detection. However, these algorithms have a low performance in small object detection. So, we produce the Expanding receptive field YOLO (ERF-YOLO) to deal with this problem. At first, we propose an efficient block which is called expanding receptive field block (ERF-block) to capture more information in larger areas. Base on YOLOv2, we down-sample the low-level location information by ERF-block, and up-sample feature information by deconvolution. Then we further assemble these two parts together to make the prediction. After training the network on VOC dataset, we have a good result with 82.6% mAP (mean Average Precision) which is 4.0% higher than the original YOLOv2 network. Thanks to the efficient block, it takes 62fps to detect one image when the input size is 416×416, which could keep a real-time speed. In addition, we also evaluate the model on a remote sensing dataset which contains many small targets, and it also shows that ours model has a better performance.

1. Introduction
The problem of small object detection has been widely concerned in the fields of remote sensing images and unmanned aerial vehicles. The traditional detection algorithms require manually setting the prior conditions, which has a poor robustness and makes it difficult to adapt to large-scale automation requirements. Besides, the traditional detection algorithms also have a low speed and bad performance on small object detection.

In recent years, with the emergence of the CNN, object detection has a fast development. Many object detection algorithms have made a satisfied performance. One of typical representative algorithms is adopted by Fast-RCNN[13] and Faster-RCNN[11], which is a two-stage structure. At first stage, it will do the feature extraction and get the region proposals, then the proposals will be classified by the deep convolution layers. Because of the two-stage structure, it is hard to meet the real-time requirement. Then YOLO[2][3] and SSD[4] have merged these two stages into one-stage, so what you need to do is to input one image, and all the things will be done in one process, which largely improve the detection speed and precision.

Most detection algorithms appear in recent years have extracted more information by increasing the depth of the network, such as VGG[5], which leads to a high inference latency. Inception[6][7] uses multi-branch structure to design and optimize the network, and replaces large-size convolution with small-size convolution to reduce network parameter. Although all above network models have excellent performance in the field of object detection, the existing detection algorithm all don’t have an ideal performance in the detection of small targets.

There has been a lot of research on the detection of small targets. Feature pyramid networks (FPN)[8] obtain that there is less feature information at the low-level layers, but the object location is accurate. At the high-level, feature information is abundant, but the target location is rough. Therefore,
FPN fuses multiple feature graph to achieve good results. PGAN\cite{11} uses Perceptual GAN to improve the detection probability of small targets. Finding tiny faces\cite{12} enhances the small faces detection by detecting different size faces on different size feature maps, and the results are quite remarkable. DSSD\cite{14} fuses the features of high and low layers through the top-down network structure, and improves the traditional up-sampling structure to improve the detection results.

With the satisfied detection performance and real-time inference speed, YOLOv2 is an algorithm which has been used in many fields. However, limited by its network structure, it can’t do well in small object detection. So, we propose the ERF-YOLO to deal with this problem. By combining the high layer network with the low-level layer, the receptive field is enlarged and good results are obtained, and the use of efficient structure makes the network meet the real-time demand.

Therefore, the major contributions of this paper are:

1) Base on YOLOv2, we design an ERF-block to obtain more information by using multi-branch structure and expanding the receptive field.

2) We utilize the low-level location information by ERF-block and the high-level feature information by deconvolution to improve the performance of the small object detection.

2. Network structure

There are researches show that low-level layers have more location information and high-level layers have more feature information\cite{8}. Therefore, taking the information above into comprehensive consideration, we propose an ERF-YOLO algorithm based on YOLOv2. We utilize the abundant location information of low-level layers by ERF-block and feature information of high-level information by deconvolution to create a new layer which both have the location and feature information\cite{10} as shown in figure 1.

![Figure 1. ERF-YOLO structure. Fusing the low-level location information and high-level feature information.](image1)

We also design a new block ERF-block to have a good performance in small object detection. Inspired by Inception\cite{6,7}, RFB\cite{15} and ASPP\cite{9}, the ERF-block is a multi-branch layer with different
efficient convolution kernel sizes, and expands the receptive field through the dilated convolution. At first, we use the 1×1, to compress the network, and then like inceptionv2, using the convolution of 1×3 and 3×1 to replace the 3×3 convolution to reduce the network parameters. Then the 3×3 and rate = 2 convolution is used to increase receptive field. The ERF-block is shown in the figure 2. This kind of convolution kernel uses the combination of dilated convolution and standard convolution to enhance the detection of the central region as shown in figure 3.

In addition, both Max pooling and avg pooling will make a serious information loss. Therefore, in order to prevent this thing happens, we use a convolution with stride equals 2 to replace the pooling layer. This method can effectively reduce the information loss caused by the pooling layer. As usual, each convolution is followed by Batchnorm and leaky-ReLU activation function.

As shown in figure 4, we use the same backbone as the YOLOv2, that is darknet-19. And then, we make some difference on the structure. At first, we use the EH-block to down-sample the Conv_4 layer, which is 104×104 and have many location information. Then we use the deconvolution to up-sampling the Conv_12 layer, which is 26×26 and have many feature information. Then we concatenate these two results and put it into the EH-Block to down-sample and get a result of 13×13. At last, we put the result and the original network Conv_19 together to make the predict.

Figure 4. The structure of the ERF-YOLO. ERF-block is used to down-sample, Deconvolution is used to up-sample.

3. Experiments

3.1. Training and testing on Pascal VOC

After designing the network, we train the YOLOv2 network at first, using the same method as the original YOLOv2. In this process, we implement the network with Tensorflow-1.10 and CUDA-9.2. We use the VOC-2007 and VOC-2012 dataset to train the network, it is a widely used dataset which contains 20 categories. We set the batch size at 32, IoU (Intersection over Union) at 0.5, which means the predicted box is positive if it’s IoU is higher than 0.5. As mentioned in YOLOv2, we also use the multi-scale training during the whole process to make the network be robust to dealing with images of different sizes. The formulation of bounding boxes and classes scores are also same as the YOLOv2. Loss function and optimization method is the same as the original network. The exponential sliding average method is used to adjust the weight parameters to get better value.

Then the pre-trained model parameters are loaded into our network. For fair comparison, we use the same environment as the YOLOv2. We still use the same dataset to fine-tuning the network. After training, we use the VOC-2007 test set to get the mAP value to evaluate the network performance. The mAP is a widely used which comprehensively consider the precision and recall of the network to evaluate the performance of the neural network. We compare the performance of our model with that several state-of-art CNN-models. As shown in table 1, our ERF-YOLO has a mAP of 82.6% which is much better than the original YOLOv2, and even has a better performance compared with other...
mainstream networks. With the improvement of accuracy, the inference time can reach 62fps and still meet the real-time requirements.

| Detection frameworks        | Train | mAP  | FPS |
|-----------------------------|-------|------|-----|
| Faster R-CNN VGG-16[1]      | 2007+2012 | 73.2 | 7   |
| SSD500[4]                   | 2007+2012 | 76.8 | 19  |
| RFB-Net512[15]              | 2007+2012 | 82.2 | 38  |
| YOLOv2 544×544[3]           | 2007+2012 | 78.6 | 40  |
| ERF-YOLO 416×416            | 2007+2012 | 81.1 | 62  |
| ERF-YOLO 544×544            | 2007+2012 | 82.6 | 39  |

As shown in table 2, we compare the performances of ERF-YOLO and some other detection algorithms on several different categories on the VOC-2012 test set. It shows that our model has a better performance, especially on small and dense objects such as boats and ships.

Table 2. Part of comparison on different categories. We list some algorithms comparison on VOC-2012, it shows that our structure has a better performance than most of others.

| method             | mAP  | Aero | Bird | Boat | Bottle | Car | Cat | Chair | Table | Plant | Sofa |
|--------------------|------|------|------|------|--------|-----|-----|-------|-------|-------|------|
| Faster-RCNN[1]     | 70.4 | 84.9 | 74.3 | 53.9 | 49.8   | 75.9| 88.5| 45.6  | 55.3  | 40.1  | 60.9 |
| SSD300[4]          | 72.4 | 85.6 | 70.5 | 57.6 | 46.2   | 76.1| 89.2| 53.0  | 60.8  | 45.9  | 69.5 |
| YOLOv2 544[3]      | 73.4 | 86.3 | 74.8 | 59.2 | 51.8   | 76.5| 90.6| 52.1  | 58.5  | 49.2  | 62.4 |
| ERF-YOLO 544       | 76.2 | 87.1 | 78.3 | 64.1 | 57.6   | 76.2| 90.3| 55.5  | 60.3  | 56.1  | 64.3 |

3.2. The performance on self-built dataset

Figure 5. remote sensing dataset. On the left, it’s the big size objects. In the middle, it’s the medium. And on the right, it’s the small size objects.

We also evaluate the model on a dataset with many small objects. As shown in figure 5, the objects in the dataset can be divided into three sizes: large, medium and small. We divide the dataset into test dataset and training dataset, and the training dataset has a proportion of 70%. Then, we train several networks and compares their performance on test set. The evaluation results are shown in table 3. It shows that ERF-YOLO has a dramatically advantage when detecting on small size objects. And the detection results of ERF-YOLO can be seen on figure 6.

Table 3. the performance of several methods on remote sensing dataset.

| method             | mAP  | Large | Medium | Small |
|--------------------|------|-------|--------|-------|
| Faster-RCNN[1]     | 66.3 | 84.2  | 69.6   | 45.1  |
| YOLOv2 544[3]      | 69.4 | 87.2  | 73.8   | 47.3  |
| SSD513[4]          | 68.0 | 85.9  | 71.4   | 46.6  |
| DSSD[4]            | 70.7 | 88.1  | 74.3   | 49.8  |
| ERF-YOLO 544       | 74.3 | 87.5  | 75.2   | 60.2  |
Figure 6. Detection results. We just list two typically results, and because the target is so small so we don’t mark the object classes on the picture.

4. Conclusion
In this paper, we introduce a real-time object detection algorithm ERF-YOLO, which is optimized for small object detection based on YOLOv2. We design the ERF-block, which obtains more information by expanding the receptive field. Then we down-sample the low-level information through the ERF-block to get more location information, and up-sample the high-level information through deconvolution to get more feature information. By combining these two sampling results, we get the detection results. We evaluate the model by VOC dataset and self-built dataset which contains many small objects, and the results show that our network has a better performance compared with those existing top-performing deeper networks. Moreover, due to the efficient network block, the structure still meets the real-time requirement.

References
[1] Ren S, He K, Girshick R, et al (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6): 1137-1149.
[2] Redmon, Joseph, et al (2016). You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas: IEEE :779-788.
[3] Redmon J, Farhadi A (2017). YOLO9000: Better, Faster, Stronger. IEEE Conference on Computer Vision and Pattern Recognition : 6517-6525.
[4] Liu W, Anguelov D, Erhan D, et al (2016). SSD: Single Shot MultiBox Detector. European Conference on Computer Vision. Amsterdam: Springer International Publishing :21-37.
[5] Simonyan K, and Zisserman A (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arxiv:1409.1556.
[6] Szegedy C, Liu W, Jia Y, et al (2014). Going deeper with convolutions. arXiv preprint arxiv: 1409.4842.
[7] Szegedy C , Vanhoucke V , Ioffe S , et al (2016). Rethinking the Inception Architecture for Computer Vision. arXiv preprint arxiv:1512.00567.
[8] Lin T, Y Dollár, Girshick R, et al. Feature pyramid networks for object detection (2017). Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition, 2017:2117-2125.
[9] Maoke Y, Kun Y, Chi Z, et al (2018). DenseASPP for Semantic Segmentation in Street Scenes. Computer Vision & Pattern Recognition, 3684-3692.
[10] Shen Z, Shi H, Feris R, et al (2017). Learning object detectors from scratch with gated recurrent feature pyramids. arXiv preprint arxiv:1712.00886v1.
[11] Li J, Liang X, Wei Y, et al (2017). Perceptual generative adversarial networks for small object detection. arXiv preprint arxiv:1706.05274.

[12] Hu P, Ramanan D (2017). Finding tiny faces. arXiv preprint arxiv: 1612.04402.

[13] Girshick R (2015). Fast R-CNN[C]. arXiv preprint arxiv: 1504.08083.

[14] Fu C Y, et al (2017). DSSD: Deconvolutional single shot detector. arXiv preprint arxiv:1701.06659.

[15] Liu S, Huang D, Wang Y (2017). Receptive Field Block Net for Accurate and Fast Object Detection. arXiv preprint arxiv:1711.07767.