SCIENCE CHINA
Information Sciences

• **Supplementary File** •

MDSSD: Multi-scale Deconvolutional Single Shot Detector for Small Objects

Lisha CUI, Rui MA*, Pei LV, Xiaoheng JIANG, Zhimin GAO, Bing ZHOU & Mingliang XU

Center for Interdisciplinary Information Science Research, Zhengzhou University, Zhengzhou 450000, China

Appendix A Related Work

Most of the traditional object detectors are based on hand-crafted features, such as Haar [1] and DPM [2]. With the development of ConvNets in recent years, the accuracy and inference speed of object detection [3–7] have been greatly improved by integrating feature learning and classifier into one framework. We classify these works based on ConvNets into the following three categories:

Detectors based on the top-most feature maps. OverFeat [3] applies a sliding window to the feature maps to create bounding boxes. It decomposes the detection into localization and classification, which tends to be costly. SPPnet [8] designs the Spatial Pyramid Pooling layer where the input images of any sizes are feasible, which is efficient in computation. R-CNN [9] and Fast R-CNN [4] use selective search to generate bounding boxes and then classify them by SVM. Faster R-CNN [10] introduces RPN (Region Propose Network) to directly generate the anchor boxes with different scales and aspect ratios on the feature maps, and thus improves effectiveness and efficiency.

YOLO [11] divides the feature maps into n regions, and then regresses and classifies the bounding boxes in each region at real-time speed. However, all these methods are based on the top-most feature maps of the convolutional neural network to locate and classify the objects. They rely on the information extracted by the upper features to a large extent and do not make full use of the bottom details.

Detectors based on multi-scale feature maps. To make full use of the multifarious information from different convolution layers to cover the objects with different scales and shapes, a set of approaches [12–17] make predictions on multi-scale feature maps. SSD [12], a single shot detector, makes predictions from multiple feature maps at different depths from bottom to top to naturally handle objects of various sizes. It is one of the state-of-the-art detectors considering both accuracy and speed. MS-CNN [13] proposes a framework which consists of a proposal sub-network and a detection sub-network. In the proposal sub-network, detection is performed at multiple output feature maps. [16] applies multiple layers derived from the backbone network for detection. M2Det [17] proposes a Multi-Level Feature Pyramid Network (MLFPN) to enhance the effectiveness of feature pyramids for detecting objects of different scales. The deconvolution layer is introduced to upsample feature maps and add context information in these methods. Nevertheless, the layers from the bottom of a ConvNet have weak semantic information, which harms their representational capacity for small object recognition.

Detectors based on connections of multi-scale feature maps. In order to enhance the representational capacity of feature maps, a number of approaches [18–22] concatenate multi-scale feature maps of ConvNets to increase context information. DSSD [23] applies successive deconvolution layers to the top of SSD to realize upsampling and then achieves connections with convolution layers. The recent methods, FPN [24] and TDM [25], adopt a top-down pathway and conduct skip connections in their architectures to enhance the power of features. Sun *et al.* [26] design a FishNet by combing features from different resolutions or depths. Libra R-CNN [27] utilizes the refining balanced semantic features at different resolutions for object detection. These algorithms [25–27], however, have cumbersome architectures which apply bottom-up and top-down pathway within the network, failing to achieve real-time processing.

Inspired by these researches, we propose MDSSD for small object detection. It combines high-level and low-level feature maps through Fusion Block to add context information for small object detection. Different from other fusion-based detectors, our Fusion Blocks are applied to multiple high-level feature maps which still remain fine details for small instances, rather than the top-most layer of the ConvNet.

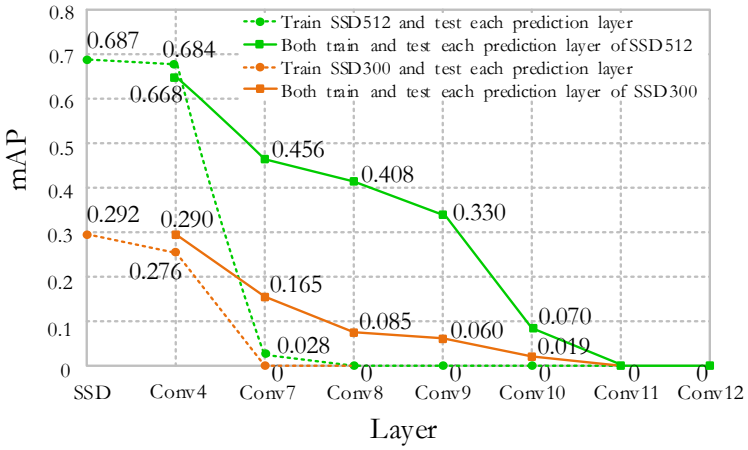


Figure B1 The detection results of SSD model on TT100K using different training and testing policies. The mAP in both ways dramatically declines to zero when the convolution layers go deeper and the feature maps get smaller, meaning that top convolutional layers are incapable of recognizing small objects, especially the last two layers.

Appendix B Methodology

Appendix B.1 Performance Analysis of Multi-scale Feature Maps

To figure out the effect of feature resolution on small object detection, we conduct pilot experiments on the benchmark TT100K with SSD model. SSD utilizes the pyramidal feature hierarchy within a ConvNet to predict objects at different scales. Predictions at multiple scales improve the mAP and single-shot architecture achieves real-time requirements. However, it is hard for SSD to detect small objects owing to the weak semantic information on shallow feature maps.

First, we fine-tune the overall SSD512 and SSD300 model where predictions are produced on several layers at different resolutions. As depicted in Figure B1, SSD512 and SSD300 achieve 68.7% and 29.2% mAP on TT100K respectively. Then by utilizing the well-trained models, we test each prediction layer separately. The mAPs of conv4 are 68.4% for SSD512 and 27.6% for SSD300, respectively. Nevertheless, the mAPs of layers after conv7 are zero, which means that shallow layers are mainly responsible for detecting small objects due to their rich fine details, and deep layers contribute little to the results of small object detection.

To further demonstrate the observation above, we fine-tune each prediction layer of SSD model independently on TT100K and test each corresponding layer. As shown in Figure B1, the mAPs of these prediction layers (from conv4 to conv12) progressively drop until zero for both SSD512 and SSD300 models. That is to say, the representational capacity of small objects becomes worse and worse along with the increasing depth of network until totally lost on the coarse, semantic deeper layers (after conv11). Inspired by the above observation, we intend to make full use of the shallow layers with rich fine details and the relatively deep layers with semantic information as well as some fine details of small objects.

Appendix B.2 Fusion Block

There are three Fusion Modules at different depths in total. We take Module 1 as an example. Figure B2 illustrates the structure of Module 1 for the 300 × 300 input model. Module 2 and Module 3 follow the same structures except the channel number. The Fusion Modules are of strong feature representational power of small instances. It is noteworthy that these high-level feature maps utilized in Fusion Block preserve both strong semantic information and some fine details of small instances, rather than the top-most layer where the representation of fine details of small objects are potentially wiped out. As for the 512×512 input model, there are some tiny modifications. Table B1 sketches the structure details of Fusion Block with 300×300 and 512×512 input.

Appendix B.3 Training

Data Augmentation. The data augmentation strategies utilized in SSD are also applied in our framework for building a robust model. In the latest version of SSD, a "zoom out" operation is implemented to improve the small object detection. We exploit both the original images and generated samples by randomly expanding and cropping for training. Please refer to SSD for more details.

Default Boxes. For Fusion Module 2 and Module 3 in MDSSD, the scales and aspect ratios of default boxes are consistent with conv4_3 and conv7 in SSD, respectively. To be specific, the scale of Module 2 and conv11 are set to 0.2 and 0.9, respectively. For a given scale, default boxes with aspect ratios of 1, 2, 3, $\frac{1}{2}$, and $\frac{1}{3}$ are generated to better match the object shape. For Module 2, conv10_2, and conv11_2, each cell of the feature maps predicts four default boxes. Others have six default boxes per each location. As for default boxes generated by Module 1, we keep them the same with Module 2

* Corresponding author (email: maruifirst@zzu.edu.cn)

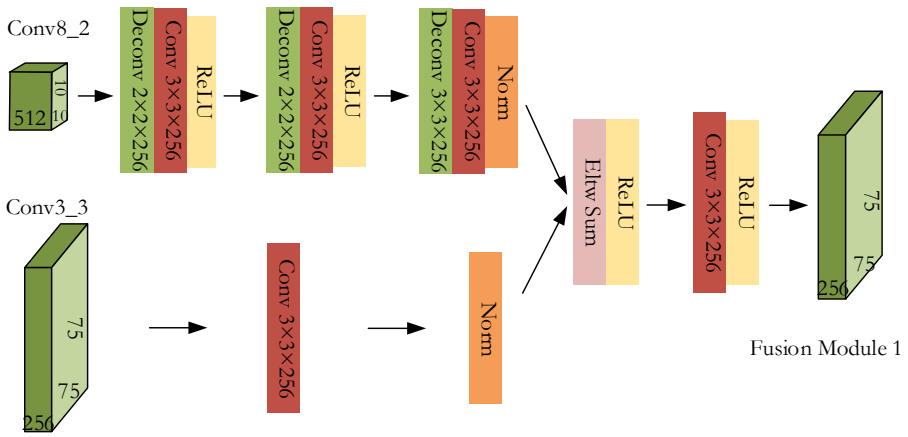


Figure B2 The architecture of Fusion Block. The top layer, i.e. Conv8_2, undergoes 3 deconvolution layers to achieve upsampling, each followed by one convolution layer. Then we apply one convolution layer with stride 1 to the bottom layer Conv3_3. The outputs of both branches are of the same size and integrated into one by element-wise summation. Finally, we obtain Fusion Module 1 after one convolution and Relu layer.

Table B1 The detailed structure of three Fusion Blocks with 300 × 300 and 512 × 512 input. [·]×2 indicates double identical operations. We set the stride to 2 for all deconvolution layers and 1 for convolution layers

Fusion Module	Module 1		Module 2		Module 3	
Connection Layers	conv3_3	conv8_2	conv4_3	conv9_2	conv7	conv10_2
Structure 300×300	3×3×256 Conv L2 Norm	[2×2×256 Deconv 3×3×256 Conv Relu] ×2 3×3×256 Deconv 3×3×256 Conv L2 Norm	3×3×512 Conv L2 Norm	[2×2×256 Deconv 3×3×256 Conv Relu] ×2 2×2×512 Deconv 3×3×512 Conv L2 Norm	3×3×1024 Conv L2 Norm	[2×2×512 Deconv 3×3×512 Conv Relu] ×2 3×3×1024 Deconv 3×3×1024 Conv L2 Norm
Structure 512×512	3×3×256 Conv L2 Norm	[2×2×256 Deconv 3×3×256 Conv Relu] ×2 2×2×256 Deconv 3×3×256 Conv L2 Norm	3×3×512 Conv L2 Norm	[2×2×256 Deconv 3×3×256 Conv Relu] ×2 2×2×512 Deconv 3×3×512 Conv L2 Norm	3×3×1024 Conv L2 Norm	[2×2×512 Deconv 3×3×512 Conv Relu] ×2 2×2×1024 Deconv 3×3×1024 Conv L2 Norm
Fusion	Eltw-sum Relu 3×3×256 Conv Relu		Eltw-sum Relu 3×3×512 Conv Relu		Eltw-sum Relu 3×3×1024 Conv Relu	

both in scale and aspect ratio. Following the strategy in SSD, we add extra conv12_2 for the 512×512 input model to make predictions due to its larger input images.

Matching and Hard Negative Mining. We match each ground truth box to the default box with the best jaccard overlap. Then we match the remaining default boxes to any ground truth with Jaccard overlap higher than a threshold (0.5). This strategy is beneficial to predict multiple bounding boxes with high scores for overlapped objects. The negative samples with top loss value are selected from the non-matched default boxes so that the ratio of positive and negative samples is 1:3.

Loss Function. The training objective is the weighted sum between localization loss (Smooth L1 [4]) and confidence loss (Softmax). More details can be found in [12].

Appendix C Experiments

We first evaluate MDSSD on small object dataset TT100K, and then the benchmark datasets PASCAL VOC2007 and MS COCO. All the experiments are implemented on Caffe platform on the machine with two 1080Ti GPUs. We use the model pre-trained on ImageNet for initialization, and then fine-tune our model on TT100K, PASCAL VOC, and MS COCO. The performance is measured by mean average precision (mAP) and we compare the results with state-of-the-art detectors about the mAP and inference speed.

Table C1 Comparisons of mAP and AP for each class on TT100K. Faster R-CNN [10], SSD512, and RFB Net are based on VGG16, while Faster R-CNN [29], FPN and Mask R-CNN utilize the deeper network ResNet101. MDSSD512 outperforms both VGG-based and ResNet-based detectors. The proposed method, RFBNet, and SSD train the models with the input size 512×512, while other approaches have larger input size 1000 × 600 or 1000 × 800

Method	mAP	i2	i4	i5	il100	il60	il80	io	ip	p10	p11	p12	p19	p23	p26	p27	p3	p5	p6	pg	ph4	ph4.5	ph5
Faster [10]	52.9	44	46	45	41	57	62	41	39	45	38	60	59	65	50	79	48	57	75	80	68	58	51
Faster [29]	61.1	59.3	73.8	79.7	76.6	76.3	68.5	64.9	66.8	52.2	58.5	45.9	48.2	74.4	66.1	64.3	65.4	74.9	39.1	78.2	58.0	36.5	69.1
FPN [24]	69.9	72.5	79.6	88.3	**90.2**	88.2	84.9	77.4	75.8	62.7	75.9	60.2	53.7	75.8	76.0	84.8	71.6	79.2	43.5	79.9	50.6	51.0	72.2
Mask R-CNN [30]	70.8	71.4	85.6	**89.0**	89.4	86.3	82.3	78.0	77.6	59.6	**76.9**	63.8	52.0	72.9	**81.7**	87.5	**78.5**	78.9	48.3	88.5	63.9	58.1	75.5
SSD512 [12]	68.7	70.1	79.3	85.3	77.1	86.4	78.7	72.3	71.6	64.5	57.1	67.7	73.0	80.4	70.7	76.2	66.5	74.9	63.9	84.2	62.1	51.2	78.6
RFBNet512 [31]	74.4	75.6	79.4	87.9	87.4	**89.9**	**88.4**	77.2	79.0	66.1	66.9	71.1	72.8	83.4	74.9	79.8	69.0	77.6	68.8	**88.9**	67.6	**63.0**	76.3
MDSSD512	**77.6**	**84.7**	**88.3**	88.8	82.1	88.1	78.4	**84.1**	**83.0**	**68.4**	75.7	**76.3**	**82.3**	**85.9**	**81.7**	**90.2**	69.4	**85.8**	**73.2**	85.4	**73.7**	59.4	**81.4**

Method	pl100	pl120	pl20	pl30	pl40	pl5	pl50	pl60	pl70	pl80	pm20	pm30	pm55	pn	pne	po	pr40	w13	w32	w55	w57	w59	wo
Faster [10]	68	67	51	43	52	53	39	53	61	52	61	**67**	61	37	47	37	75	33	54	39	48	39	37
Faster [29]	77.6	74.6	40.5	48.5	60.2	65.4	49.0	51.2	61.2	59.0	50.5	29.1	68.5	77.8	87.5	47.7	86.9	30.9	57.2	62.1	67.0	57.2	42.7
FPN [24]	87.5	85.5	55.7	55.6	71.5	77.3	60.8	58.7	63.5	70.9	55.5	40.1	75.7	**89.0**	89.8	60.2	87.6	45.3	67.8	65.9	70.3	62.3	**53.2**
Mask R-CNN [30]	86.7	82.4	58.6	53.3	68.2	76.4	63.5	56.6	66.3	71.5	58.0	41.5	68.8	88.6	**90.5**	63.0	87.5	51.3	60.6	66.6	71.1	61.8	47.6
SSD 512 [12]	85.1	84.2	45.4	66.6	65.7	60.5	58.3	64.0	**70.6**	70.5	69.6	51.3	71.2	71.7	86.4	51.8	87.9	46.1	57.1	64.6	74.0	58.8	39.7
RFB Net 512 [31]	**88.8**	84.9	**66.8**	71.8	71.6	75.0	62.9	70.4	64.9	71.9	**73.7**	54.0	**86.5**	78.0	88.2	59.8	84.5	64.8	70.1	72.4	81.5	69.3	43.7
MDSSD512	84.9	**88.0**	66.6	**73.9**	**76.4**	**78.5**	**70.7**	**72.1**	70.4	**79.2**	72.7	49.5	77.5	85.4	88.2	**68.1**	**90.0**	**67.0**	**73.3**	**78.9**	**82.1**	**80.6**	50.1

Table C2 The detection results over multi-scale small objects which are less than 100×100 pixels in 2048×2048 images. 'A' refers to the areas of objects, where the area is measured as the number of pixels in the segmentation mask. 'P' denotes the percentage of objects in the images

Object Size	$0<A \leqslant 20^2$	$20^2<A \leqslant 40^2$	$40^2<A \leqslant 60^2$	$60^2<A \leqslant 100^2$
Percentage	$0<P \leqslant 0.98\%$	$0.98\%<P \leqslant 1.95\%$	$1.95\%<P \leqslant 2.93\%$	$2.93\%<P \leqslant 4.88\%$
Accuracy	43.4%	75.0%	81.6%	90.9%

Appendix C.1 Results on TT100K

TT100K is a traffic-sign benchmark dataset where target objects typically occupy a very small proportion of each image. For instance, a traffic sign may be only 20 × 20 pixels in a 2048 × 2048 image. Therefore, we only evaluate the MDSSD model with 512 × 512 input, termed MDSSD512. Following [28], we only reserve the categories with more than 100 instances, which leaves 45 classes of traffic signs to detect. MDSSD512 model is trained on the original training set (including 6105 images) and evaluated on the test set (including 3071 images). We train MDSSD512 with the initial learning rate of 10^{-3} for the first $200k$ iterations, then decrease it to 10^{-4} for the next $100k$ iterations and 10^{-5} for another $40k$ iterations. Because of the relatively large images in TT100K, the batch size is set to 8 considering the GPU memory. The momentum and weight decay are set to 0.9 and 0.0005 respectively by using SGD.

Table C1 demonstrates the results of MDSSD512 compared with other object detectors. Faster R-CNN [10], SSD512, and RFB Net are based on VGG16, while Faster R-CNN [29], FPN and Mask R-CNN utilize the deeper network ResNet101. Additionally, Faster R-CNN [10,29], FPN, and Mask R-CNN are trained on Detectron [1)], and we change the scale of anchors from [32^2, 64^2, 128^2, 256^2, 512^2] to [16^2, 32^2, 64^2, 128^2, 256^2] for better matching the small traffic signs in TT100K.

It can be observed that MDSSD512 achieves an mAP of 77.6%, outperforming SSD512 (68.7% mAP) and RFB Net (74.4 % mAP) by 8.9 and 3.2 points respectively with the same input size of 512 × 512 and the same backbone network. Moreover, MDSSD512 can also exceed the two-stage detectors Faster R-CNN [10,29] (52.9% and 61.1% mAP) with a large margin, even though they have larger input size of 1000 × 600. FPN and Mask R-CNN, the variants of Faster R-CNN, apply ResNet101 as the backbone network and perform well on small object detection. They achieve 69.9% and 70.8% mAP, which are still inferior to the proposed model. The results on TT100K demonstrate the effectiveness of MDSSD for small object detection.

To further analyze MDSSD over different object sizes on TT100K, we test 1000 images using the well-trained model. The object instances smaller than 100×100 pixels in 2048×2048 images are categorized into 4 classes (i.e., $(0, 20^2]$, $(20^2, 40^2]$, $(40^2, 60^2]$, and $(60^2, 100^2]$) according to their areas, where the area is measured as the number of pixels in the segmentation mask. We calculate the percentage of these small objects which are all less than 5% of the images as well, as displayed in Table C2. The accuracies of the 4 classes are 43.4%, 75.0%, 81.6% and 90.9%, respectively. The accuracy of the first class is less than satisfactory because the objects are particularly small, occupying less than 1% of an image. These objects can hardly be found even by naked eyes. As the objects get larger, the accuracy increases dramatically.

Appendix C.2 Results on PASCAL VOC2007

To validate the performance of MDSSD for general object detection, we train MDSSD on PASCAL VOC2007 and VOC2012 `trainval` (16551 images), and test on VOC2007 `test` (4952 images). Batch size is set to 32 for 300×300 input and 16 for

1) https://github.com/facebookresearch/Detectron

Table C3 Detection results on PASCAL VOC2007 `test` set. All the methods are trained on VOC2007 `train` and VOC2012 `trainval`, and tested on VOC2007 `test`. SSD300* and SSD512* indicate the latest version updated by the authors

Method	Backbone	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Faster [10]	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
ION [19]	VGG	75.6	79.2	83.1	77.6	65.6	54.9	85.4	85.1	87	54.4	80.6	73.8	85.3	82.2	82.2	74.4	47.1	75.8	72.7	84.2	80.4
Faster [29]	ResNet-101	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
MR-CNN [13]	VGG	78.2	80.3	84.1	78.5	70.8	68.5	88.0	85.9	87.8	60.3	85.2	73.7	87.2	86.5	85.0	76.4	48.5	76.3	75.5	85.0	81.0
R-FCN [32]	ResNet-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* [12]	VGG	77.5	79.5	83.9	76.0	69.6	50.5	87.0	85.7	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* [12]	VGG	79.5	84.8	85.1	81.5	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
DSSD321 [23]	ResNet-101	78.6	81.9	84.9	80.5	68.4	53.9	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	87.2	79.4
DSSD513 [23]	ResNet-101	**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
MDSSD300	VGG	78.6	86.5	87.6	78.9	70.6	55.4	86.9	87.0	88.3	58.5	84.8	73.4	84.8	89.2	88.1	78.0	52.4	78.6	74.5	86.8	80.7
MDSSD512	VGG	80.3	**88.8**	88.7	**83.2**	73.7	58.3	88.2	89.3	87.4	62.4	85.1	75.1	84.7	89.7	88.3	83.2	**56.7**	84.0	77.4	83.9	77.6
MDSSD320*	ResNet-101	79.1	85.3	88.1	78.0	64.1	45.3	89.4	89.3	**93.4**	60.3	80.4	74.2	90.2	91.4	86.6	81.4	50.4	79.0	83.9	91.0	79.5
MDSSD512*	ResNet-101	81.0	87.0	**89.9**	81.6	63.7	46.9	**92.7**	**91.8**	92.8	61.7	83.6	73.9	**91.3**	**94.2**	**88.8**	**84.8**	52.1	80.4	**86.6**	**94.0**	**82.8**

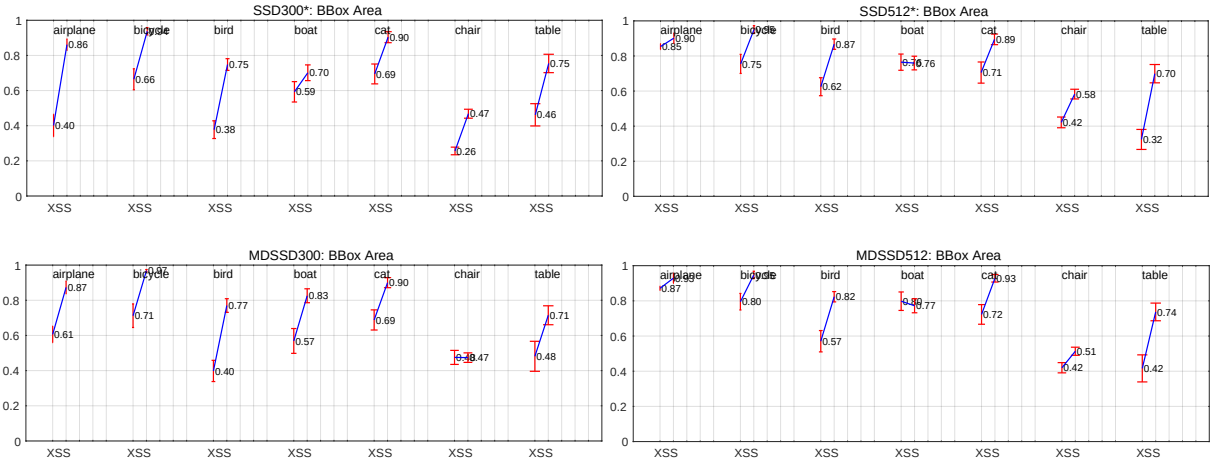


Figure C1 Sensitivity and impact of object size on VOC2007 `test` set using [33], including 7 object categories. The top row shows the effects of BBox Area per category for the latest SSD300* and SSD512* model respectively, and the bottom row shows our results. Key: BBox Area: XS=extra-small; S=small.

512×512 input. We use 10^{-3} learning rate for the first $60k$ iterations, then decrease it to 10^{-4} for the next $40k$ iterations and 10^{-5} for another $20k$ iterations.

Table C3 shows our detection results on VOC2007 `test` compared with other state-of-the-art architectures. Our model with 300×300 input has achieved 78.6% mAP. It exceeds the latest SSD300* by 1.1 points and can be comparable to DSSD321 with 321×321 input. By increasing the image size to 512×512, MDSSD achieves better performance, improving mAP from 79.5% to 80.3%. The mAP of DSSD513 is higher than that of MDSSD512, but we argue that this is because DSSD utilizes ResNet-101 as the backbone network. However, it should be noted that MDSSD512 is much faster than DSSD513 (see Table C6).

For a fair comparison with other methods, we replace the backbone from VGG16 to ResNet-101 for training MDSSD on VOC2007 as well, denoting as MDSSD320* and MDSSD512*. As shown in Table C3, MDSSD320* (79.1%) and MDSSD512* (81.0%) achieve better results than the original models, improving MDSSD300 and MDSSD512 by 0.5 and 0.7 points respectively. Additionally, MDSSD320* yields 0.5 points gain compared with DSSD321, while MDSSD512* is slightly inferior to DSSD513. The results demonstrate that MDSSD is more effective for small input size. Nevertheless, the deep backbone network hampers the inference speed of MDSSD in the meantime, as shown in Table C6.

In order to verify the performance of MDSSD on small object detection, we also utilize the detection analysis tool from [33]. Following the definition of [33], each object is assigned to a size category depending on the object's percentile size within its category: extra-small (XS: bottom 10%); small (S: next 20%); medium (M: next 40%); large (L: next 20%); extra-large (XL: next 10%). To clearly demonstrate the improvement for small object detection, we only show the results of category XS and S.

Figure C1 shows the comparison between our methods and SSD for sensitivity and impact of object size, including 7 object categories. For better visualization, we calculate mAP of the 7 categories for S and XS, as shown in Table C4. MDSSD300 achieves 56% mAP and 79% mAP for category XS and S, respectively. Our models improve baseline SSD300* with 49% mAP and 77% mAP by 7 and 2 points, respectively. The mAPs of category XS and S are 66% and 81% for MDSSD512, while 63% and 81% for SSD512*. That is to say, our MDSSD300 model shows significant improvement compared with SSD300* model, while MDSSD512 brings marginal performance gain compared with SSD512* model. This performance proves that the proposed MDSSSD is much more effective for detecting small objects. The AP of some specific

Table C4 Comparison of mAP between SSD and MDSSD for object XS and S, which includes 7 object categories shown in Figure C1. MDSSD300 shows significant improvement compared with SSD300* model, while MDSSD512 brings slightly performance gain compared with SSD512* model. The results demonstrate that the proposed MDSSSD is much more effective for small input size.

Method	mAP(%)	
	XS	S
SSD300*	49	77
SSD512*	63	81
MDSSD300	56	79
MDSSD512	66	81

Table C5 The comparison of detection results among different methods on COCO `test-dev2015`. MDSSD improves the original SSD models with a large margin. It is obvious that MDSSD models achieve the highest AP (the 5^{th} column) and AR (the 7^{th} column) for S, which demonstrates the effectiveness of MDSSD for small object detection

Method	Data	Backbone	Avg.Precision, IoU:			Avg.Precision	Avg.Recall, #Dets:			Avg. Recall
			0.5:0.95	0.5	0.75	Area: S	1	10	100	Area: S
Faster [10]	trainval	VGGNet	21.9	42.7	-	-	-	-	-	-
ION [19]	train	VGGNet	23.6	43.2	23.6	6.4	23.2	32.7	33.5	10.1
Faster [29]	trainval	ResNet-101	34.9	55.7	37.4	15.6	-	-	-	-
R-FCN [32]	trainval	ResNet-101	29.9	51.9	-	10.8	-	-	-	-
YOLOv2 [21]	trainval35k	Darknet-19	21.6	44.0	19.2	5.0	20.7	31.6	33.3	9.8
SSD300* [12]	trainval35k	VGGNet	25.1	43.1	25.8	6.6	23.7	35.1	37.2	11.2
SSD512* [12]	trainval35k	VGGNet	28.8	48.5	30.3	10.9	26.1	39.5	42.0	16.5
DSSD321 [23]	trainval35k	ResNet-101	28.0	46.1	29.2	7.4	25.5	37.1	39.4	12.7
DSSD513 [23]	trainval35k	ResNet-101	33.2	53.3	35.2	13.0	28.9	43.5	46.2	21.8
DSOD300 [15]	trainval	DS/64-192-48-1	29.3	47.3	30.6	9.4	27.3	40.7	43.0	16.7
MDSSD300	trainval35k	VGGNet	27.8	46.0	28.7	10.8	24.3	36.6	38.8	15.8
MDSSD512	trainval35k	VGGNet	31.1	52.5	33.4	13.9	27.3	42.3	45.9	22.4

classes improves significantly as well, such as airplane with the background of the sky. This may benefit from the Fusion Modules with context information.

Appendix C.3 Results on MS COCO

To further validate our model, we train MDSSD300 and MDSSD512 on MS COCO as well. We use the `trainval35k` (118287 images) for training and evaluate the results on the standard `test-dev2015` split (20288 images). The batch size is set to 16 for 300×300 input and 8 for 512×512 input. We train the model with 10^{-3} for the first $160k$ iterations, then 10^{-4} and 10^{-5} for another $120k$ and $40k$ iterations.

MS COCO defines that the objects are small (area $< 32^2$), medium ($32^2 <$ area $< 96^2$), large (area $> 96^2$), where the area is measured as the number of pixels in the segmentation mask. To obtain results on COCO `test-dev2015` where the ground-truth annotations are hidden, we upload generated results to the evaluation server to get the performance analysis.

In Table C5, we observe that MDSSD300 achieves 27.8% AP@[0.5:0.95], 46.0% AP@0.5, and 28.7% AP@0.75, which improves the conventional SSD300* by 2.7, 2.9, and 2.9 points, respectively. MDSSD512 also outperforms the baseline SSD512* by 2.3, 4.0, and 3.1 points, respectively. Even though our models do not perform as well as DSSD, it should be noticed that the backbone network of MDSSD is VGG16 and MDSSD is about 4 times faster than DSSD. Compared with the other detectors based on VGG16 such as Faster R-CNN and ION with 1000 × 600 input, MDSSD achieves the best results.

It is noticeable that our MDSSD300 and MDSSD512 model achieve 10.8% AP and 13.9% AP for small objects (area $<32^2$), respectively. Our models improve SSD (6.6% and 10.9%), DSSD (7.4% and 13%), and DSOD (9.4%/-) with a large margin. MDSSD outperforms all one-stage architectures based on both VGG16 and ResNet-101. Our method achieves a higher AR (average recall) for small objects as well, which proves that MDSSD is more powerful for small object detection.

Appendix C.4 Inference Time

New parameters need to be learned due to additional layers in MDSSD, therefore the inference speed of the network being hampered. We use 2000 images with batch size 1 to evaluate the inference speed of MDSSD on a machine with one 1080Ti GPU. The results are presented in the 5th column of Table C6, including other state-of-the-art methods. For a fair comparison, we verify SSD on the same single Nvidia 1080Ti GPU as well. MDSSD300 and MDSSD512 model run at 38.5 FPS and 17.3 FPS, respectively. Although the speed is a little lower than SSD, it can still meet the real-time application.

Table C6 The comparison of Speed and Accuracy on PASCAL VOC2007 dataset. All the methods are trained on the union of VOC2007 and VOC2012 `trainval` and tested on VOC2007 `test`

Method	Backbone	GPU	Input Size	speed(FPS)	mAP(%) VOC2007
Faster [10]	VGG16	Titan X	$\sim$ 1000×600	7	73.2
Faster [29]	ResNet-101	K40	$\sim$ 1000×600	2.4	76.4
R-FCN [32]	ResNet-101	Titan X	$\sim$ 1000×600	9	80.5
SSD300* [12]	VGG16	Titan X	300×300	46	77.5
SSD512* [12]	VGG16	Titan X	512×512	19	79.8
DSSD321 [23]	ResNet-101	Titan X	321×321	9.5	78.6
DSSD513 [23]	ResNet-101	Titan X	513×513	5.5	81.5
DSOD300 [15]	DS/64-192-48-1	Titan X	300×300	17.4	77.7
MDSSD300	VGG16	1080Ti	300×300	38.5	78.6
MDSSD512	VGG16	1080Ti	512×512	17.3	80.3
MDSSD320*	ResNet-101	1080Ti	320×320	14.7	79.1
MDSSD512*	ResNet-101	1080Ti	512×512	9.3	81.0

Our method exceeds the two-stage networks with a large margin in terms of speed, and it can also outperform one-stage methods DSSD and DSOD.

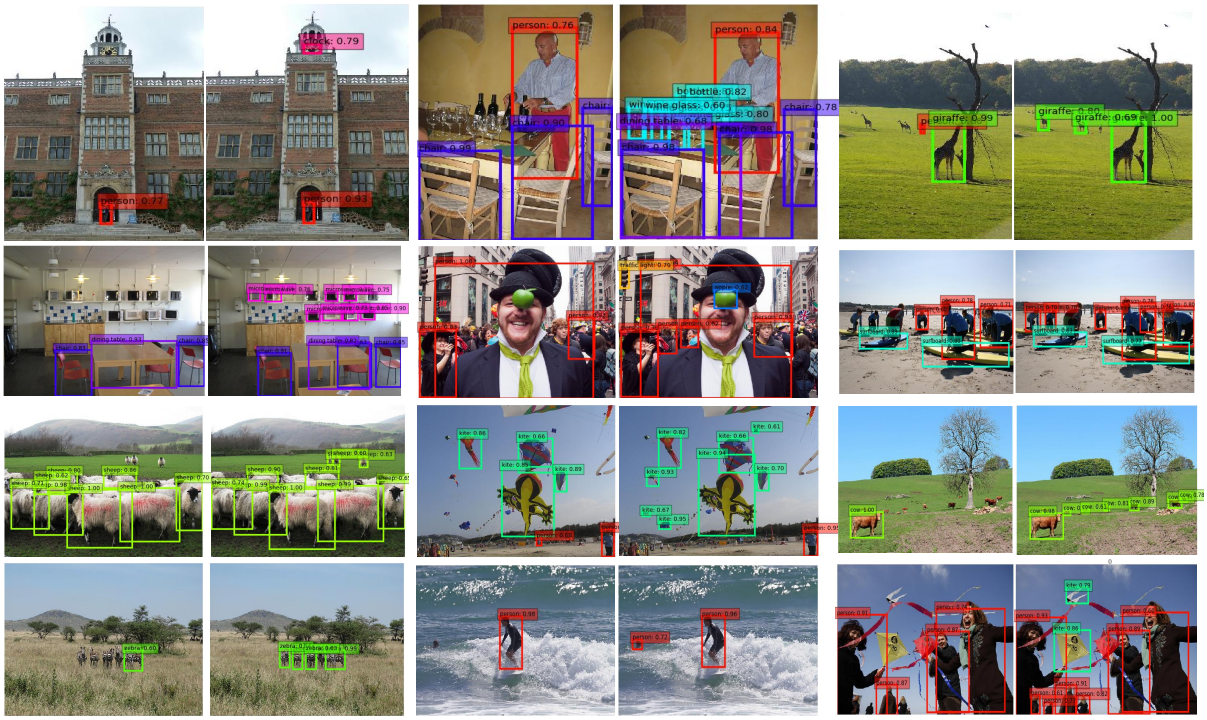
The speeds of MDSSD320* and MDSSD512* are 14.7 and 9.3 FPS respectively, which declines dramatically compared with MDSSD300 and MDSSD512. However, they are still faster than DSSD321 (9.5 FPS) and DSSD513 (5.5 FPS) which are based on ResNet-101 as well. It is mainly because we only implement connections for the shallow prediction modules instead of every prediction layer.

Appendix C.5 Visualization Results

Figure C2 illustrates some visualization results on the TT100K test set. Figure C3 shows some detection results of MDSSD compared with SSD on VOC2007 `test` and COCO `test-dev2015`. We only display the bounding boxes with a score higher than 0.6. Different colors of the bounding boxes indicate different object categories. MDSSD can detect the very small objects on TT100K, though some of them are difficult to distinguish by naked eyes. As for VOC and COCO, our model performs better than conventional SSD in most cases. We can observe that MDSSD yields better performance on small and occluded objects both in Figure C3(a) and C3(b).

Figure C2 The detection results on the TT100K test set. The objects are tightly surrounded by the bounding boxes with the categories and classification scores on them. We only show the detections with scores higher than 0.6. We zoom in the detection results marked by the white rectangles for better visualization. Each color corresponds to an object category. MDSSD can detect the very small traffic signs on TT100K, though some of them are difficult to distinguish by naked eyes.

(a) The detection results in scenes containing small or occluded objects on PASCAL VOC2007 `test` set



(b) The detection results in scenes containing small or occluded objects on COCO `test-dev2015` set

Figure C3 The detection results of MDSSD (column 2, column 4, and column 6) and SSD (column 1, column 3, and column 5) in scenes containing small or occluded objects. We can see that MDSSD yields better performance on small and occluded objects both in (a) and (b).

References

1 Viola P A, Jones M J. Rapid object detection using a boosted cascade of simple features. In: Computer Vision and Pattern Recognition, Kauai, 2001. 511–518
2 Felzenszwalb P F, Girshick R B, Mcallester D, et al. Object detection with discriminatively trained part-based models. IEEE Trans Pattern Anal Mach Intell, 2014, 47: 6–7
3 Sermanet P, Eigen D, Zhang X, et al. Overfeat: Integrated recognition, localization and detection using convolutional networks. In: International Conference on Learning Representations, Banff, 2014.
4 Girshick R B. Fast r-cnn. In: International Conference on Computer Vision, Santiago, 2015. 1440–1448
5 Geng Q C, Zhou Z, Cao X C. Survey of recent progress in semantic image segmentation with CNNs. Sci China Inf Sci, 2018, 61: 051101
6 Liu X L, Hou F, Qin H, et al. A CADe system for nodule detection in thoracic CT images based on artificial neural network. Sci China Inf Sci, 2017, 60: 072106
7 Liu S H, Wang S Q, Shi W H, et al. Vehicle tracking by detection in UAV aerial video. Sci China Inf Sci, 2019, 62: 024101
8 He K M, Zhang X Y, Ren S Q, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition. In: European Conference on Computer Vision, Zurich, 2014. 346–361
9 Girshick R B, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Computer Vision and Pattern Recognition, Columbus, 2014. 580–587
10 Ren S Q, He K M, Girshick R B, et al. Faster r-cnn: towards real-time object detection with region proposal networks. In: International Conference on Neural Information Processing Systems, Montreal, 2015. 91–99
11 Redmon J, Divvala S K, Girshick R B, et al. You only look once: Unified, real-time object detection. In: Computer Vision and Pattern Recognition, Las Vegas, 2016. 779–788
12 Liu W, Anguelov D, Erhan D, et al. Ssd: Single shot multibox detector. In: European Conference on Computer Vision, Amsterdam, 2016. 21–37
13 Cai Z W, Fan Q F, Feris P S, et al. A unified multi-scale deep convolutional neural network for fast object detection. In: European Conference on Computer Vision, Amsterdam, 2016. 354–370
14 Zhao J P, Guo W W, Zhang Z H, et al. A coupled convolutional neural network for small and densely clustered ship detection in SAR images. Sci China Inf Sci, 2019, 62: 042301
15 Shen Z Q, Liu Z, Li J G, et al. Dsod: Learning deeply supervised object detectors from scratch. In: International Conference on Computer Vision, Venice, 2017. 1937–1945
16 Tian Z, Shen C H, Chen H, et al. FCOS: Fully Convolutional One-Stage Object Detection. In: International Conference on Computer Vision, Seoul, 2019
17 Zhao Q J, Sheng T, Wang Y T, et al. M2Det: A Single-Shot Object Detector Based on Multi-Level Feature Pyramid Network. In: Conference on Artificial Intelligence, Honolulu, 2019, 9259–9266
18 Kong T, Yao A B, Chen Y R, et al. Hypernet: Towards accurate region proposal generation and joint object detection. In: Computer Vision and Pattern Recognition, Las Vegas, 2016. 845–853
19 Bell S, Zitnick C L, Bala K, et al. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In: Computer Vision and Pattern Recognition, Las Vegas, 2016. 2874–2883
20 Liu W, Rabinovich A, Berg A C. Parsenet: Looking wider to see better. arXiv preprint arXiv:1506.04579, 2015.
21 Redmon J, Farhadi A. Yolo9000: Better, faster, stronger. In: Computer Vision and Pattern Recognition, Honolulu, 2017. 6517–6525
22 Honari S, Yosinski J, Vincent P, et al. Recombinator networks: Learning coarse-to-fine feature aggregation. In: Computer Vision and Pattern Recognition, Las Vegas, 2016. 5743–5752
23 Fu C Y, Liu W, Ranga A, et al. Dssd : Deconvolutional single shot detector. 2017. arXiv preprint arXiv:1701.06659
24 Lin T Y, Dollr P, Girshick R B, et al. Feature pyramid networks for object detection. In: Computer Vision and Pattern Recognition, Honolulu, 2017. 936–944
25 Shrivastava A, Sukthankar R, Malik J, et al. Beyond skip connections: Top-down modulation for object detection. CoRR, vol. abs/1612.06851, 2016.
26 Sun S Y, Pang J M, Shi J P, et al. FishNet: A Versatile Backbone for Image, Region, and Pixel Level Prediction. arXiv: http://arxiv.org/abs/1901.03495, 2019
27 Pang J M, Chen K, Shi J P, et al. Libra R-CNN: Towards Balanced Learning for Object Detection. In: Conference on Computer Vision and Pattern Recognition, Long Beach, 2019, 821–830
28 Zhu Z, Liang D, Zhang S H, et al. Traffic-sign detection and classification in the wild. In: Computer Vision and Pattern Recognition, Las Vegas, 2016. 2110–2118
29 He K M, Zhang X Y, Ren S Q, et al. Deep residual learning for image recognition. In: Computer Vision and Pattern Recognition, Las Vegas, 2016. 770–778
30 He K M, Gkioxari G, Dollár P, Mask R-CNN. In: International Conference on Computer Vision, Venice, 2017. 2980–2988
31 Liu S T, Huang D, Wang Y H, Receptive field block net for accurate and fast object detection. In: European Conference on Computer Vision, Munich, 2018. 404–419
32 Dai J F, Li Y, He K M, et al. R-fcn: Object detection via region-based fully convolutional networks. In: Advances in Neural Information Processing Systems, Barcelona, 2016. 379–387
33 Hoiem D, Chodpathumwan Y, Dai Q Y. Diagnosing error in object detectors. In: European Conference on Computer Vision, Florence, 2012. 340–353