StairsNet: Mixed Multi-scale Network for Object Detection

Weiyi Gao[1,2(✉)], Wenlong Cao[1,2], Jian Zhai[2], and Jianwu Rui[2]

[1] University of Chinese Academy of Sciences, Beijing, China
{weiyi,wenlong}@nfs.iscas.ac.cn
[2] Institute of Software, Chinese Academy of Sciences,
National Engineering Research Center of Fundamental Software, Beijing, China

Abstract. It is common to choose image classification network as backbone in the object detector. The art-of-the-state image classification network exhibits excellent performance on image classification, but that network hurts the detection efficiency, mainly due to the coarseness of features from several convolution and pooling layers. In this paper, we present a single deep neural network with inceptions, called ***StairsNet***, to take advantage of the art-of-the-state image classification network in object detection. In contrast to previous single network SSD [13] which uses VGG-16 as a feature to extract network, our approach applies recently state-of-the-art classification network Residual Network (ResNets [5]). Meanwhile, to avoid coarseness of the last CNN feature, StairsNet not only utilizes various of scale features, but also mixes different scale features to predict. To this end, we insert two stairs-like architectures into the network: top stairway network that mixes multi-scale feature maps as input to predict bounding boxes and bottom stairway network that turns into two different scale feature branches. Our StairsNet significantly increases the PASCAL-style mean Average Precision (mAP) from 75.0% (SSD + ResNet-101) to 77.7%. Code is available at https://github.com/gwyve/caffe/tree/StairsNet.

Keywords: StairsNet · Object detection · ResNet · Inception

1 Introduction

With the recent advance of convolution neural networks (CNNs) [6], a great progress has been made these years on image classification [5,9,20,22] and object detection [4,13,17,24]. Since R-CNN [4] was established by Girshick *et al.*, many recent detectors followed this paradigm: firstly, a object proposal algorithm [26] generates candidate regions; secondly, the CNNs classify every proposed region. Faster R-CNN [17] replaces conventional object proposal algorithm with a new CNN RPN (Region Proposal Networks). On the other hand, following MultiBox [3,23], a much faster detector, SSD [13], utilizes a single network to predict bounding box.

B. Zeng et al. (Eds.): PCM 2017, LNCS 10735, pp. 303–314, 2018.
https://doi.org/10.1007/978-3-319-77380-3_29

Recently, many modern object detectors are changing the extract feature layers with the change of state-of-the-art image classification networks. It is tempting to focus on adding recent state-of-the-art image classification networks such as Residual Network (ResNets [5]) and GoogLeNet [22] into the object detection architecture. However, the combination of a state-of-the-art classifier (ResNet-101 [5]) and SSD doesn't show dramatic performance in improving accuracy, as the phenomenon that inferior detection accuracy doesn't match the network's superior classification accuracy. Dai *et al.* [10] argued that the issue is caused by lack of respecting variance for object detection and inserted position-sensitive score maps into the framework to remedy that issue. However, it applies the expensive RPN to generate proposal regions in R-FCN [10], as in Faster R-CNN [17].

To make SSD with ResNet-101 more excellent detector, we not only append ResNet block to the extract feature layers, but also insert Inception-style architecture [7,11,21,22] into the network to predict the bounding boxes and classify the object. Inspired by Inception [21,22,25], we use inception to capture the nonlinear concepts from CNN feature map. Meanwhile, the last CNN feature map is too coarse to detect some small-size objects. Although SSD uses six scales to detect the object, we support the method that combining multi-scale features before predicting networks can improve ability to detect small-size objects [12]. Motivated by that fact, we mix multi-scale features to predict. Each inception located at each scale pipeline captures the nonlinear concepts from single scale CNN feature map before combination.

In this paper, we develop a object detection framework called ***StairsNet***. Our network consists of the 101-layer Residual Net (ResNet-101) as the backbone and several stair modules. The stairs networks are divided into two stairways: top stairway network that mixes multi-scale feature maps as input to predict bounding boxes and bottom stairway network that turns into two different scale feature branches. When being evaluated, our StairsNet significantly increases the PASCAL-style mean Average Precision (mAP) from 75.0% (SSD + ResNet-101) to 77.7%. Code is available at https://github.com/gwyve/caffe/tree/StairsNet.

2 Related Work

Object detection is one of the fundamental tasks in computer vision and rapid progress recently. Many researches follow R-CNN, a two-stage detection paradigm: Firstly, an object algorithm generates candidate regions that may contain an object; Secondly, the CNNs classify each proposed region. Before using deep networks, the majority of proposal algorithms include those based on grouping super-pixels (*e.g.* MCG [14], CPMC [2], Selective Search [26]) and those based on sliding windows (*e.g.* EdgeBoxes [27], objectness in windows [1]). In the Faster R-CNN paper, the Selective Search region proposals, which are based on low-level image features, are replaced by the ones learned from a region proposal network (RPN). The YOLO approach by Redmon *et al.* [15] uses a single network to predict bounding boxes and class probabilities, in an end to end network. It divides the input image into a grid of cells and predicts the coordinates

and confidences of objects contained in the cells. Adopting one-stage detection, the YOLO is much faster than Faster R-CNN. Similar with YOLO, SSD also adopts a single network in that. The difference is that SSD uses a fixed set of default boxes for prediction, which is like the anchor in RPN.

Multi-scale: Overfeat [19] classifies different scale boxes. In recent MultiBox works [3,23], there are multi-scale features from different convolution layers to be used as input. The SSD adopts six scale features to predict, in which progress we use the combination of two of those scale features. Additionally, HyperNet [8] combines all the feature of different scales with deconvolution, different from the method we only combine two scale feature.

Inception-Style Network: Following the Network in Network (NIN) [11], Szegedy *et al.* [22] inserted inception in pipeline. They verified that approximating the expected sparse structure-Inception by readily available dense building blocks is a viable method for improving neural networks for detection. Szegedy *et al.* [25] provides several design principles in the context of the Inception. Based on the principles, various types of Inception architecture have been presented. As the introduction of residual connection [21], training with residual connection accelerates the training of Inception networks significantly.

3 Model

Our object detection system is named as **StairsNet**, a stairs-like architecture. The entire system is a single, unified network for object detection, which use ResNet-101 as backbone. StairsNet is divided into two flights of stairs (Top-Stairs and Bottom-Stairs) by landing. Each stair tread of Top-Stairs has two inceptions (Up-Inception and Down-Inception) and features-combined module in StairsNet. Figure 1 is an overview of the system. In Sect. 3.1 we illustrate the details about using Residual-101 as backbone. In Sect. 3.2 we introduce the two modules of StairsNet and we demonstrate different inception-style architecture in Sect. 3.3. In Sect. 3.4 we show the details about training strategy for StairsNet. In Sect. 3.5 we show some training details.

3.1 Using Residual-101 as Backbone

Single Shot MultiBox Detector (SSD [13]) chooses truncated VGG-16 [20] as base network, and appends extra convolution layers to base network. Each of the added layers is used to predict scores and offsets for some predefined default bounding boxes. With dramatic success of ResNet in image classification, using ResNet as backbone in the object detector becomes a prevalent method. To use the advantage of ResNet, we replace VGG-16 with Residual-101. Additionally, we append additional four residual blocks (Res6_x, Res7_x, Res8_x, Res9_x) after Res5_x block which is the end of Residual-101's full-convolution layers as Table 1 shows. Because of the mismatch of residual block and the input size in original SSD paper, we change the input size to 321×321. Simply doing those doesn't improve accuracy and therefore we present StairsNet modules in our framework.

Table 1. Architectures ResNet-101 vs. StairsNet. We replace VGG-16 with Residual-101. Additionally, we append additional four residual blocks (Res6_x, Res7_x, Res8_x, Res9_x) after Res5_x which is the end of Residual-101's full-convolution layers.

SSD layer	ResNet 101	StairsNet	StairsNet layer	Output size
conv1	7×7, 64, stride 2	7×7, 64, stride 2	Res1_x	161×161
conv2_x	3×3 max pool, stride 2	3×3 max pool, stride 2	Res2_x	80×80
	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$		
conv3_x	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	**Res3_x**	40×40
conv4_x	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	Res4_x	20×20
conv5_x	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	**Res5_x**	20×20
conv6_x		$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 512 \end{bmatrix} \times 1$	**Res6_x**	10×10
conv7_x		$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 512 \end{bmatrix} \times 1$	**Res7_x**	5×5
conv8_x		$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 512 \end{bmatrix} \times 1$	**Res8_x**	3×3
conv9_x		$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 512 \end{bmatrix} \times 1$	**Res9_x**	1×1

3.2 StairsNet

Similar with U-shaped stairs, StairsNet is divided into two *flights* of stairs by *landing* .The first module is set of Inception-Style networks named **Top-Stairs**. The second module named **Bottom-Stairs** is one-by-one corresponded by *newel* with the first module every stairstep.

Bottom-Stairs includes 5 stairsteps as the color pink block showed in Fig. 1. We use state-of-the-art classification model ResNet-101 neural network in place of VGG-16 in SSD to get the more discriminable image feature. The *tread* of upstairs is the different residual layers of ResNet-101. More details are showed in Sect. 3.1.

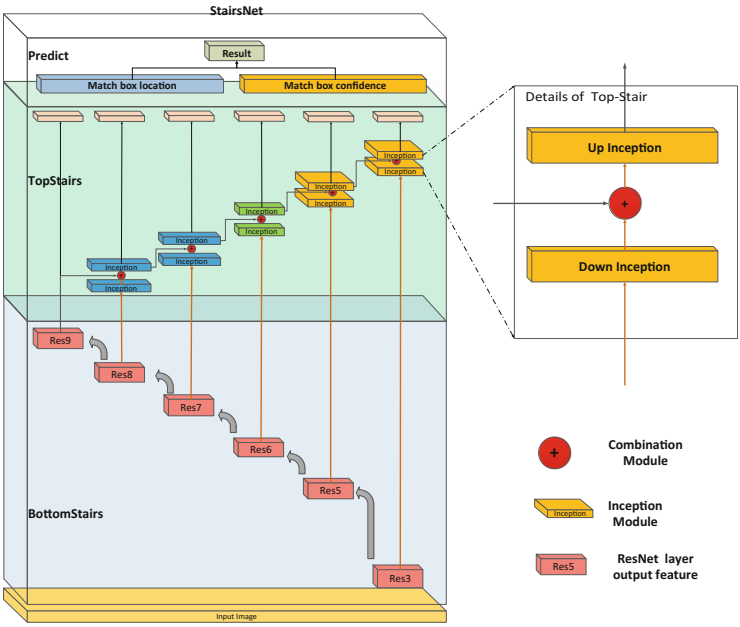

Fig. 1. StairsNet network architecture. The red layers of Bottom-Stairs are Res-block as extra layers. Each stair tread of Top-Stairs has two inceptions and combination module. The *newel* connected Top-Stairs and Bottom-Stairs one-by-one is the residual layer feature for predictor. (Color figure online)

Landing is where a 180° change in direction is made. In our system, the *landing* is where the feature transform direction changes, *e.g.* Res9_x in Fig. 1. After the *Landing*, the input data not only comes from the previous layers, but also combines the deconv information from the successor. In Fig. 1 red circle is combination mixing different scale features.

(**Top-Stairs**) module of StairsNet is used to get location proposals and to match bounding boxes. One stairstep has two inception blocks and a features-combined module. The inception of each stairsteps changes depending on the difference of feature scale. Top-stairs uses three different inceptions which are showed in Fig. 2. In Fig. 1, downwards, the first two represents Inception-A, the middle of inceptions represents Inception-B, and the last two represents Inception-C. More details about inceptions we use is showed in next section. The combination module changes the large scale to small scale by deconvolution, and then uses Eltwise layer to mix the two feature maps. As is shown in color green block in Fig. 1, some *newel* connects Bottom-Stairs and Top-Stairs one-by-one is the residual layer feature for predictor, and the stairstep's input is the combination of different scale features.

3.3 Inception

SSD [13] uses convolution filter as a bounding box proposal and classification module. We argue that, as a generalized linear model (GLM) for the underlying data patch, the level of abstraction of the convolution layer is low. As described in [11], GLM can achieve great result when the samples of the latent concepts are linearly separable, and the data for the same concepts are generally highly nonlinear function of the input. Therefore, the representations that capture these concepts are generally highly nonlinear function of the input. Inception [11] has been a successful nonlinear function module in neural network design [22]. To utilize the added computation as efficiently as possible, Szegedy *et al.* [25] explored methods to scale up networks by suitably factorized convolutions and aggressive regularization. In our model, in place of convolution in SSD, we use Fig. 2(a), (b) and (c) to capture these highly nonlinear feature from different scales for next stage. Additionally, we use inception for detecting relatively small and large objects simultaneously in location proposal module.

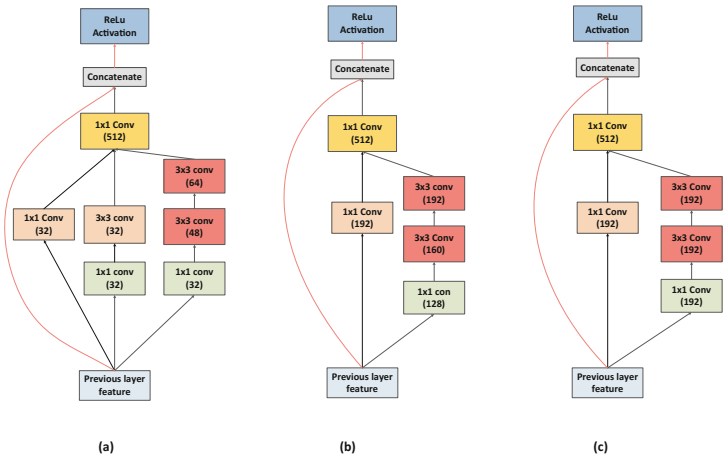

Fig. 2. Inception Module. (a) is Inception-A used for Res3_x's and Res5_x's Top-Stairs. (b) is Inception-B used for Res6_x's Top-Stairs and (c) is Inception-C used for Res7_x's and Res8_x's Top-Stairs.

3.4 Optimizing Strategy

Similar to SSD, the StairsNet objective is extending MultiBox objective [3,23] to handle multiple object categories. During training, we are selecting from default boxes of different location at each location in several feature maps with different scales, with using matching strategy in SSD which is for each groundtruth box. For the i-th default bounding box($d_i = (d_i^{cx}, d_i^{cy}, d_i^w, d_i^h)$), we predict the offsets to the j-th groundtruth box($g_j = (g_j^{cx}, g_j^{cy}, g_j^w, g_j^h)$),

$$\hat{g}_j^{cx} = \frac{(g_j^{cx} - d_i^{cx})}{d_i^w} \qquad \hat{g}_j^{cy} = \frac{(g_j^{cy} - d_i^{cy})}{d_i^h} \tag{1}$$

$$\hat{g}_j^{w} = \log(\frac{g_j^{w}}{d_i^{w}}) \qquad \hat{g}_j^{h} = \log(\frac{g_j^{h}}{d_i^{h}}) \tag{2}$$

and the predict location (l_i) will be get the confidences for all object categories. The overall model loss is a weighted (λ) sum of the localization loss (loc) and the confidence loss ($conf$).

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \lambda L_{loc}(x, l, g)) \tag{3}$$

where N is the number of matched default boxes, $x_{ij}^p = \{0, 1\}$ be an indicator for matching the i-th default box to the j-th groundtruth box of category p.

$$L_{loc}(x, l, g) = \sum_{i \in Positive}^{N} \sum_{m \in \{cx,cy,w,h\}} x_{ij}^k Smooth\_L_1(l_i^m - \hat{g}_j^m) \tag{4}$$

$$L_{conf}(x, c) = -\sum_{i \in Positive}^{N} x_{ij}^p \log(\hat{c}_i^p) - \sum_{i \in Negative} \log(\hat{c}_i^0) \tag{5}$$

where $\hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}$, and the λ is set to 1 and the positive default bounding box is matched with groundtruth box object and the rest as negatives.

3.5 Training

In contrast to SSD, StairsNet has a small change in the prior box aspect ratio setting. The original SSD model, boxes with aspect ratios of 2 and 3 were proven useful from the experiments. Similar with [16], we run k-means clustering on the training boxes with square root of box area as the feature. Results show that most of boxes ratios fall within a range of 1–3. Therefore, we add 1.6 to the aspect radios, and use total three aspect radios at each prediction layer.

The key difference between training SSD is that we train the model including separated 2 steps. Firstly, we build and train base network using Residual-101 classification model, which is pre-trained on the ILSVRC CLS-LOC dataset [18], as backbone instead of VGG-16. Secondly, frozing the the backbone network (Residual-101) parameter, we add the Bottom-stairs and Top-stairs into the architecture and train the entire StairsNet. We apply the same network architecture we used for VOC2007 + 2012 dataset. When training the base network, we first train the model with 10^{-3} learning rate for 40k iterations, and then continue training for 20k iterations with 10^{-4} and 10^{-5}. Lastly, we continue training our model StairsNet for 20k iterations with learning rating 10^{-3}, 10^{-4} and 10^{-5}, with utilizing the trained base network model. We can achieve 77.7% mAP on the VOC2007 test set.

4 Experiments

4.1 Model Result on VOC2007

We compare against Faster R-CNN, SSD, StairsNet on VOC2007 `test` (4950 images). All methods are trained on VOC2007 + 2012 `train`. Figure 1 shows the architecture details of the StairsNet321 model. The results show StairsNet have better effect on object detection and large size input will increase result (Table 2).

Table 2. Pascal VOC2007 `test` detection result. Faster R-CNN use input images whose minimum dimension is 600. The SSD models have exactly the same settings except different input sizes (300 × 300 vs. 512 × 512). SSD+ResNet321 is that ResNet-101 instead of VGG-16 of SSD and input size is 321 × 321. The two StairsNet models have same settings except that they have different input sizes (321 × 321 vs. 513 × 513).

Method	Dataset	Base network	Using inception	mAP
Faster R-CNN	VOC2007	VGG-16	N	73.2%
SSD300	VOC2007	VGG-16	N	77.2%
SSD512	VOC2007	VGG-16	N	79.8%
SSD+ResNet321	VOC2007	ResNet-101	N	**75.0%**
StairsNet321	VOC2007	ResNet-101	Y	**77.7%**
StairsNet513	VOC2007	ResNet-101	Y	80.1%

4.2 Inference Time

We measure the speed with batch size 1 using NVIDIA GTX 1080Ti and cuDNN v5 with Intel(R) Core(TM) i7-6700K CPU@4.00GHz. Table 3 shows the comparison between StairsNet, Faster R-CNN [17], SSD [13]. Our proposed model is not fast as the SSD for two reasons. Firstly, the ResNet-101 is slower than VGGNet. Secondly, the extra layers we added to the model introduce extra overhead. Both our StairsNet321 and StairsNet513 method outperforms Faster R-CNN in both speed and accuracy. Our StairsNet513 model has better accuracy, but is slightly slower. Therefore, using a faster base network could even further improve the speed, which can possibly make the StairsNet513 model real-time as well.

4.3 Model Analysis

To understand the affects of each component in StairsNet, we carry out controlled experiments. In all the experiments, we use the same settings and input size (321 × 321) in specified changes to the settings or component(s).

Do Inceptions Help?
Each Top-Stairs module has two inceptions at the both sides of the deconvolution. Closer to the output, the Up-Inception can capture highly nonlinear features after being mixed by deconvolution. As Table 4 shows, the model can't

Table 3. Pascal VOC2007 `test` detection results. StairsNet321 is real-time detection method that can achieve above 77.7% mAP. By using a larger input image, Stairs513 outperforms all methods on accuracy.

Method	mAP	FPS	Batch size	#Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	14	1	~ 6000	$\sim 1000 \times 600$
SSD300	77.2	92	1	8732	300×300
SSD512	79.8	38	1	24564	512×512
StairsNet321	77.7	20	1	17080	321×321
StairsNet513	80.1	13	1	43936	513×513

find any detection unless using Up Inception. This fact shows Up-Inception has a tremendous effect on mixed feature to predict. In Table 4, the low result when not using Down-Inception shows the function of Down-Inception in StairsNet.

Table 4. Effects of various design choices and components on StairsNet performance. ✱ represents that the model couldn't find any detection when evaluated.

Component name	StairsNet321				
Deconvolution		✓	✓	✓	✓
Up-Inception			✓		✓
Down-Inception				✓	✓
VOC2007 `test` mAP	75.0%	✱	77.2%	✱	**77.7%**

Can We Utilize the Same Inceptions?
In StairsNet, depending on the different scales, we utilize the three kinds of inceptions to handle different scale nonlinear features. Is it necessary to build different inceptions? Will all the same inceptions with more parameters helps? To this end, we replace all inceptions in StairsNet with the Inception-C. The mAP of the StairsNet with the same inceptions is 70.0%. The result shows that it is better to use various of inceptions depending on the different scales.

Large Size Image will Help Result?
Although increasing the size of input will spend more time to train and test, Similar with SSD, using large size image as input will increase mAP. By increasing the training and testing image size 513×513, we are 0.3% more accurate than using SSD 512×512 model.

4.4 Visualization Our Result

In Fig. 3, we show some detection examples on VOC2007 `test` with SSD300 and StairsNet321 models. Compared to SSD, the small size objects and certain classes that have distinct context have been improved.

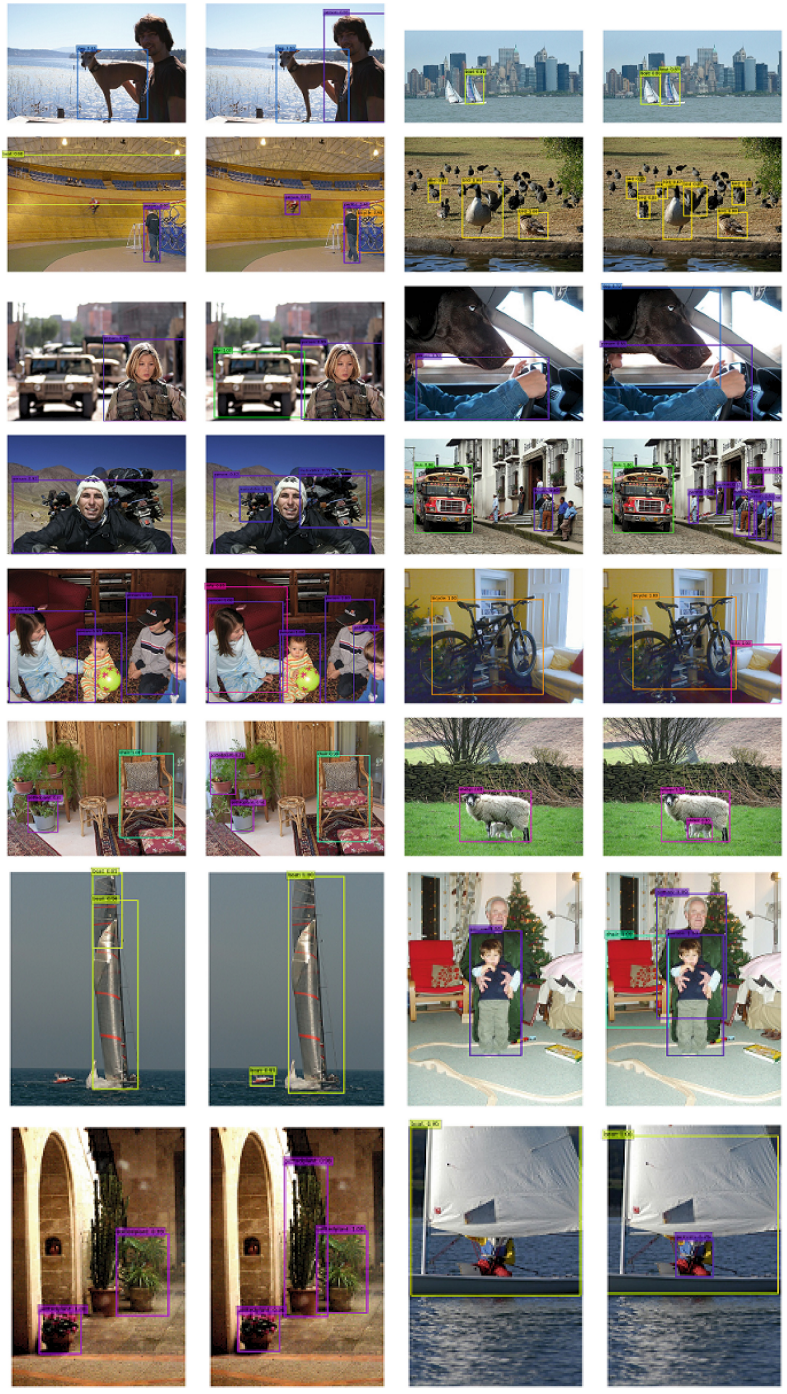

Fig. 3. Detection examples on VOC2007 `test` with StairsNet321 model. For each pair, the left side is the result of **SSD** and right side is the result of **StairsNet**. We show detection result with scores higher than 0.6. Each color corresponds to an object category.

5 Conclusions

We have presented StairsNet, a single deep neural network for object detection in images. StairsNet provides with an effective features-combined method to utilize the mixed multi-scale features to predict. Meanwhile, inceptions added not only captures the nonlinear concepts from different scale features, but also decides the effect of mixed features. We demonstrate its effectiveness on benchmark datasets by the experiments in this paper. Using ResNet and mixing multi-scale feature, StairsNet has the better effect in object detection.

Acknowledgments. This work is supported by the National Natural Science Foundation of China under Grant No. 61432001, the National Science and Technology Major Project under Grant No. 2014ZX01029101-002, the Youth Innovation Promotion Association CAS under Grant No. 2016105, and the National High-tech R&D Program (863 Program) under Grant No. 2013AA01A603.

References

1. Alexe, B., Deselaers, T., Ferrari, V.: Measuring the objectness of image windows. IEEE Trans. Pattern Anal. Mach. Intell. **34**(11), 2189–202 (2012)
2. Carreira, J., Sminchisescu, C.: CPMC: automatic object segmentation using constrained parametric min-cuts. IEEE Trans. Pattern Anal. Mach. Intell. **34**(7), 1312–1328 (2012)
3. Erhan, D., Szegedy, C., Toshev, A., Anguelov, D.: Scalable object detection using deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2147–2154 (2014)
4. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
6. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. science **313**(5786), 504–507 (2006)
7. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift (2015) arXiv preprint arXiv:1502.03167
8. Kong, T., Yao, A., Chen, Y., Sun, F.: HyperNet: towards accurate region proposal generation and joint object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 845–853 (2016)
9. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp. 1097–1105 (2012)
10. Li, Y., He, K., Sun, J., et al.: R-FCN: Object detection via region-based fully convolutional networks. In: Advances in Neural Information Processing Systems, pp. 379–387 (2016)
11. Lin, M., Chen, Q., Yan, S.: Network in network (2013). arXiv preprint arXiv:1312.4400
12. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature Pyramid Networks for Object Detection (2016)

13. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C.: SSD: single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2
14. Pont-Tuset, J., Arbelaez, P., Barron, J.T., Marques, F., Malik, J.: Multiscale combinatorial grouping for image segmentation and object proposal generation. IEEE Trans. Pattern Anal. Mach. Intell. **39**(1), 128–140 (2017)
15. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
16. Redmon, J., Farhadi, A.: Yolo9000: Better, Faster, Stronger (2016). arXiv preprint arXiv:1612.08242
17. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems, pp. 91–99 (2015)
18. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vision **115**(3), 211–252 (2015)
19. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: OverFeat: Integrated recognition, localization and detection using convolutional networks (2013). arXiv preprint arXiv:1312.6229
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014). arXiv preprint arXiv:1409.1556
21. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, Inception-Resnet and the impact of residual connections on learning (2016). arXiv preprint arXiv:1602.07261
22. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
23. Szegedy, C., Reed, S., Erhan, D., Anguelov, D., Ioffe, S.: Scalable, high-quality object detection. Computer Science (2014)
24. Szegedy, C., Toshev, A., Erhan, D.: Deep neural networks for object detection. In: Advances in Neural Information Processing Systems, pp. 2553–2561 (2013)
25. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
26. Uijlings, J.R., Van De Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. Int. J. Comput. Vision **104**(2), 154–171 (2013)
27. Zitnick, C.L., Dollár, P.: Edge Boxes: locating object proposals from edges. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 391–405. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_26