NETNet: Neighbor Erasing and Transferring Network for Better Single Shot Object Detection

Yazhao Li[1], Yanwei Pang[1]*, Jianbing Shen[2], Jiale Cao[1], Ling Shao[2]

[1]Tianjin Key Laboratory of Brain-inspired Intelligence Technology,
School of Electrical and Information Engineering, Tianjin University, Tianjin, China
[2]Inception Institute of Artificial Intelligence, Abu Dhabi, UAE

[1]{lyztju, pyw, connor}@tju.edu.cn, [2]{shenjianbingcg@gmail.com, ling.shao@ieee.org}

Abstract

Due to the advantages of real-time detection and improved performance, single-shot detectors have gained great attention recently. To solve the complex scale variations, single-shot detectors make scale-aware predictions based on multiple pyramid layers. However, the features in the pyramid are not scale-aware enough, which limits the detection performance. Two common problems in single-shot detectors caused by object scale variations can be observed: (1) small objects are easily missed; (2) the salient part of a large object is sometimes detected as an object. With this observation, we propose a new Neighbor Erasing and Transferring (NET) mechanism to reconfigure the pyramid features and explore scale-aware features. In NET, a Neighbor Erasing Module (NEM) is designed to erase the salient features of large objects and emphasize the features of small objects in shallow layers. A Neighbor Transferring Module (NTM) is introduced to transfer the erased features and highlight large objects in deep layers. With this mechanism, a single-shot network called NETNet is constructed for scale-aware object detection. In addition, we propose to aggregate nearest neighboring pyramid features to enhance our NET. NETNet achieves 38.5% AP at a speed of 27 FPS and 32.0% AP at a speed of 55 FPS on MS COCO dataset. As a result, NETNet achieves a better trade-off for real-time and accurate object detection.

1. Introduction

With the emergence of deep neural networks [26, 43, 7], object detection built on deep networks has achieved significant progress both in detection accuracy [13, 5, 27] and detection efficiency [39, 40, 18]. Despite this success, complex scale variations in practical scenes exist as a fundamental challenge and a bottleneck for accurate object detection [44, 45, 20]. Beneficial from an optimal trade-off between real-time detection efficiency and accurate detection performance, single-shot detectors [35, 31] have gained increased popularity recently.

Exploring multi-level features is essential [36, 58, 50] to tackle complex scale variations. The single-shot detector SSD [35] has been proposed and developed based on pyramid feature representation. SSD implements scale-aware object detection by detecting different-sized objects within different layers of the pyramid, which is motivated by the fact that deep-layer features with small feature resolution contain more semantic information for large objects, while the features for small objects are found in the shallow layers with large feature resolution [24, 59]. Therefore, shallow layers are responsible for detecting small objects and

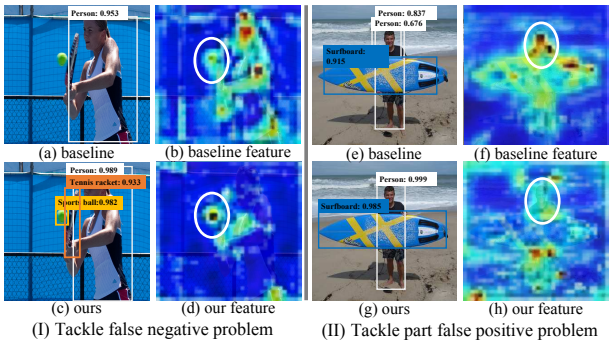


Figure 1. Two common detection problems for baseline SSD [35] and the solution using our NETNet. The visualized features are extracted from the first pyramid layer for detecting small objects. (I) False negative problem. The small objects (tennis racket, sports ball) are missed in (a) because features of small objects are not salient on the corresponding pyramid features (b). Our NETNet can detect small objects with high confidence by erasing the features of large objects and focusing on small objects as (c, d). (II) Part false positive problem. The head is detected as another person in baseline because this part region is highlighted on the features (f) for detecting small objects. Our NETNet can solve this problem by suppressing the salient part features of large objects, as (h).

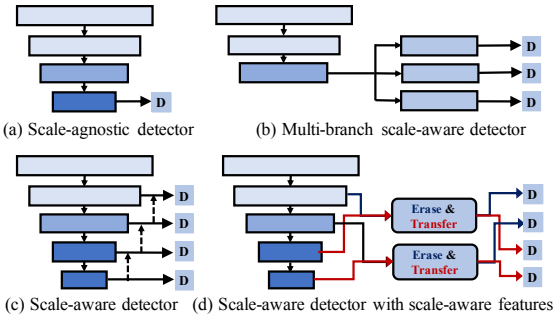
*Corresponding author: Yanwei Pang

deep layers are devoted to detecting large objects. Based on feature pyramid, some methods explore to further enhance the feature representation by fusing multi-scale features using an extra feature pyramid, which has proven useful [25, 12, 30, 31, 22] for improving detection performance. Although single-shot detectors have made great progress by adopting the feature pyramid, several failure cases, such as missing small objects and poor localization [17, 23], still exist limiting detection performance.

In most previous single-shot detectors, features are scale-confused instead of scale-aware even on one specific pyramid layer. For example, in some shallow layers of a feature pyramid, features for both small and large objects exist. As shown in Fig. 1, in the shallow features (b) used for detecting small objects, the large-object features dominate the main saliency, weakening the small-object features and thus preventing the detection of small objects (*e.g.*, the sports ball from (a) is not detected in the final result). Additionally, some parts of large objects have strong response regions on shallow features. For example, the head region in Fig. 1(e) is highlighted in (f), which leads to the wrongly detection of the head region. Thus, the features are scale-confused making it difficult to solve these two problems, *i.e.*, false negative problem and part false positive problem.

With this observation, we propose to generate scale-aware features for better single-shot object detection. To achieve this, redundant features are erased to alleviate feature scale-confusion. Thus, we only retain features of small objects in the shallow layers, erasing features of large objects. Then, we use these small-scale-aware features to detect small objects. As shown in Fig. 1(d), most of the features of large objects are removed. The features of small objects are thus emphasized, enabling the small sports ball to be detected precisely. The salient features of large objects can also be suppressed to alleviate the part false positive problem, as shown in (h). Meanwhile, transferring these erased features to a suitable scale (*i.e.*, large-scale) space could enhance the features of large objects and improve the overall detection performance.

The main contributions of our method are as follows: *We propose a Neighbor Erasing and Transferring (NET) mechanism to generate scale-aware features. Two modules, the Neighbor Erasing Module (NEM) and Neighbor Transferring Module (NTM), are designed in our NET mechanism to unmix the scale confusion and enhance feature aggregation, respectively. The NEM, embedded with a reversed gate-guided erasing procedure, is to extract and erase the large object features from the shallow layers. Then, the large object features are transferred to the deep pyramid layers by the NTM for enhancing the deep features. With our NET mechanism, a modified single-shot network, NETNet, is constructed by simultaneously embedding the scale-aware features and the scale-aware prediction. In NETNet, we enrich the pyramid features by introducing a Nearest Neighbor Fusion Module (NNFM). As a result, our NETNet is capable of achieving fast and accurate object detection with a better trade-off than previous single-shot detectors.*



(a) Scale-agnostic detector (b) Multi-branch scale-aware detector
(c) Scale-aware detector (d) Scale-aware detector with scale-aware features

Figure 2. Different detectors for object detection.

2. Related Work

Scale-agnostic detectors. Most recent object detectors are built upon deep networks. The regions with CNN features (R-CNN) methods [14, 13] integrate a CNN into object detection and achieve promising performance. As a two-stage method, the Faster R-CNN [41] proposes a lightweight network for generating proposals and construct the detection network as a complete end-to-end network. Methods like YOLO [39], Faster R-CNN [13], R-FCN [5], and other variants [29, 6, 1] have made significant progress for improving detection accuracy and efficiency. As shown in Fig. 2(a), this type of methods detect all objects of various scales by utilizing the deepest single-scale high-level features. Thus, these detectors are scale-agnostic detectors.

Scale-aware detectors. Due to the complex scale variations, many researchers have explored to exploit multi-scale pyramid features for object detection [30, 10] as well as other visual tasks [37, 48, 46, 51]. SSD [35] is a single-shot (*i.e.*, single-stage) detector that proposes to make scale-aware prediction based on multi-layer pyramid features. Features in shallow layers are used for detecting small objects and features in deep layers for large objects. RFBNet [33] embeds multi-scale receptive fields to enhance feature discriminability. DES [59] enriches the semantics of object features through a semantic segmentation branch and a global activation module. FPN [30], DSSD [10], and RONet [24] involve extra top-down feature pyramids and detect objects on each scale of these pyramids as shown in Fig. 2(c). Most recent methods [31, 21, 19, 61, 60] have explored the advantages of the pyramid features and have achieved promising results. Kong *et al*. [22] proposed to reconfigure the pyramid features by aggregating multi-layer features and reassigning them into different levels. Recent TridentNet [28] attempts to generate scale-specific features through a parallel multi-branch architectures as shown in Fig. 2(b) by embedding different receptive fields, which

achieves promising improvement on two-stage detectors.

Different from these methods, we propose to generate scale-aware features for single-shot object detection by introducing an erasing and transferring mechanism. The adversarial erasing strategy has also been investigated in weakly supervised object localization [54, 57], weakly supervised semantic segmentation [16], and salient object detection [4]. In these methods, the well recognized regions are erased to refine the prediction results iteratively. Different from them, we propose to reconfigure the pyramid features to scale-aware features by removing the scale-uncorrelated features using an erasing strategy. The erased features in shallow layers are further transferred to enhance features in deep layers, instead of discarding them as previous erasing methods. Besides, attention mechanism has been widely explored recently [53, 11, 49, 52, 9]. We introduce an attention to guide the erasing and transferring. As shown in Fig. 2(d), we aim to reconfigure the pyramid features to scale-aware features and alleviate the scale-confusion using the Erasing and Transferring operation. We then build a single-shot scale-aware detector for more accurate object detection.

3. NET Mechanism

To tackle complex scale variations, we propose to generate scale-aware features for object detection. As can be observed from Fig. 1(b) and (f), features in the shallow pyramid layers contain detailed information for both large objects and small objects. However, features for large objects are more salient, which causes small objects to be missed in Fig. 1(a) and the part false positive problem in Fig. 1(e). Instead of applying feature fusion as previous top-down feature pyramids [30, 10], we propose a NET mechanism to reconfigure the basic pyramid features to scale-aware features for scale-aware object detection. As shown in Fig. 3(a), in the NET mechanism, a feature erasing module (*i.e.*, NEM) and a feature transferring module (*i.e.*, NTM) are contained. The NEM is designed to remove large-object features from the shallow layers and emphasize the features of small objects. Then the NTM is used to transfer these features to enhance the deep features. Because our method aims to generate scale-aware features for scale-aware prediction, we take the typical single-shot detector SSD [35] as our baseline in which a pyramid from the backbone network is adopted for multi-scale prediction. We first analyze the feature pyramid in the baseline SSD. Then, we present the details of our NEM and NTM in the NET mechanism.

3.1. Basic Feature Pyramid

In SSD, a feature pyramid is explored to detect objects with different scales. We denote the objects with a specific scale s^{th} as x_s. The objects for all S scales are represented as $X = \{x_1, x_2, ..., x_S\}$, where x_1 represents objects with smallest scale and x_S refers to objects with largest scale.

SSD detects objects in a pyramidal hierarchy by exploiting multiple CNN layers, with each layer is responsible for detecting objects of a specific scale [38]. In the feature pyramid with S layers, we denote the features from s^{th} layer as p_s and express all the pyramid features as $P = \{p_1, p_2, ..., p_S\}$, where p_1 represents features with largest resolution in the shallow pyramid layer for detecting small objects x_1. With feature pooling in the pyramid, feature resolution is decreased from p_1 to p_S. Obviously, information for small objects are gradually discarded from shallow to deep layers. Because of the small input image size (*e.g.*, 300×300) for SSD, the deep layers (*e.g.*, with spatial size 5×5) only contain features for large objects. Thus, we can approximately get:

$$p_s = f_s(x_s, x_{s+1}, ..., x_S), \tag{1}$$

where $f_s(x)$ represents the feature extraction of the pyramid. The feature scale-confusion in a shallow layer (*e.g.*, p_1 contains features for various-scale objects) makes detecting small objects difficult and leads to much part detection, as shown in Fig. 1. We propose to reconfigure the pyramid features to be scale-aware features and solve these problems.

3.2. Neighbor Erasing Module

To alleviate feature scale-confusion, we propose a Neighbor Erasing Module (NEM) to filter out the redundant features. Suppose two adjacent pyramid layers, s^{th} layer and $(s+1)^{th}$ layer. Obviously, features in the s^{th} layer $p_s = f_s(x_s, x_{s+1}, ..., x_S) \in \mathbb{R}^{h_s \times w_s \times c_s}$ have more information for objects x_s than features in the $(s+1)^{th}$ layer $p_{s+1} = f_{s+1}(x_{s+1}, ..., x_S) \in \mathbb{R}^{h_{s+1} \times w_{s+1} \times c_{s+1}}$, where ($h_s > h_{s+1}$, $w_s > w_{s+1}$). Based on this feature distribution, we can generate features $\tilde{p}_s = f_s(x_s)$ for objects with scale s from the pyramid feature p_s, by erasing features $p_{es} = f_s(x_{s+1}, ..., x_S)$ of objects in a scale range of $[s+1, S]$ as:

$$\tilde{p}_s = p_s \ominus p_{es} = f_s(x_s, ..., x_S) \ominus f_s(x_{s+1}, ..., x_S), \tag{2}$$

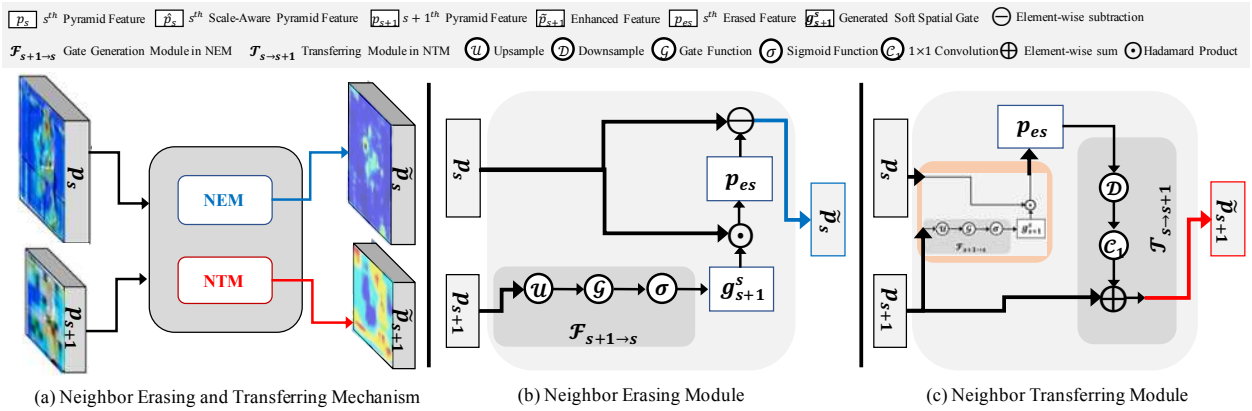
with an element-wise subtraction operation $\ominus$.

Noticing that pyramid feature p_{s+1} only contains information for objects with a scale range of $[s+1, S]$, we therefore use p_{s+1} to guide the feature erasing in Eq. 2. Specifically, we extract the feature p_{es} from p_s by:

$$p_{es} = p_s \odot \mathcal{F}_{s+1 \to s}(p_{s+1}), \tag{3}$$

where $\odot$ refers to Hadamard product. $\mathcal{F}_{s+1 \to s}(p_{s+1})$ can be represented as a soft spatial gate $g^s_{s+1} \in [0,1]^{h_s \times w_s \times c}$ (c is from $\{1, c_s\}$). We generate this gate by using the features from the $(s+1)^{th}$ pyramid layer and adopt it to guide suppressing features of objects $\{x_{s+1}, ..., x_S\}$ in p_s. In our implementation, we calculate this spatial gate as:

$$g^s_{s+1} = \mathcal{F}_{s+1 \to s}(p_{s+1}) = \frac{1}{1 + e^{-\mathcal{G}(\mathcal{U}(p_{s+1}); W^s_{s+1})}}, \tag{4}$$



(a) Neighbor Erasing and Transferring Mechanism (b) Neighbor Erasing Module (c) Neighbor Transferring Module

Figure 3. The Neighbor Erasing and Transferring (NET) mechanism (a), with (b) Neighbor Erasing Module (NEM), and (c) Neighbor Transferring Module (NTM). After NETM, $\tilde{p}_s$ highlights small objects, and deep feature $\tilde{p}_{s+1}$ contains more information for larger objects.

where $\mathcal{U}(p_{s+1})$ upsamples p_{s+1} to $p^s_{s+1} \in \mathbb{R}^{h_s \times w_s \times c_{s+1}}$ to keep the consistent spatial resolution between the gate g^s_{s+1} and feature p_s. We implement the gate function $\mathcal{G}(.)$ with learnable weights W^s_{s+1}.

In actual, since $\mathcal{G}(.)$ can be represented as a self-attention function [53] in which attention for objects can be extracted from the input features, we can construct it based on the spatial attention mechanism in [53] and [11]. Alternately, we can choose to use max pooling or average pooling along channel direction to generate a spatial attention map ($c = 1$) like that in [55] as:

$$\mathcal{G}(p^s_{s+1}) = \mathcal{P}_{max}(p^s_{s+1}) \text{ or } \mathcal{P}_{avg}(p^s_{s+1}), \quad (5)$$

or combining max pooling $\mathcal{P}_{max}(.)$ and average pooling $\mathcal{P}_{avg}(.)$ by a convolution layer with W^s_{s+1}. In our implementation, we use a $1 \times 1 \times c_s$ convolution layer $\mathcal{C}_{1\times1}$ as:

$$\mathcal{G}(p^s_{s+1}) = \mathcal{C}_{1\times1}(p^s_{s+1}; W^s_{s+1}), \quad (6)$$

to generate a channel-wise spatial gate for extracting and suppressing the features of larger objects in p_s, since it is proved an optimal trade-off between precision and efficiency as Sec. 5.1. In summary, we generate the scale-aware features $\tilde{p}_s$ for smaller objects x_s by suppressing the features of larger objects via a reversed gate as:

$$\tilde{p}_s = f_s(x_s) = p_s \ominus p_{es} = p_s \ominus (p_s \odot g^s_{s+1}). \quad (7)$$

3.3. Neighbor Transferring Module

As discussed above, in the pyramid feature p_s, some detailed information for objects $\{x_{s+1}, x_{s+2}, ..., x_S\}$ is also contained. Although this detailed information disturbs features for detecting smaller objects x_s, it is helpful for enhancing the features of larger objects x_n $(n > s)$ for more accurate classification and localization. Therefore, we propose to transfer these features from a shallow layer (*e.g.*, p_s) to a deep layer (*e.g.*, p_{s+1}).

As formulated in Sec. 3.2, the soft spatial gate g^s_{s+1} generated by p_{s+1} has larger activation values on the regions for objects $\{x_{s+1}, ..., x_S\}$. Thus, p_{es} in Eq. 3 helps extract the detailed information of these larger objects. We then transfer this detailed information p_{es} and obtain the new pyramid features $\tilde{p}_{s+1} \in \mathbb{R}^{h_{s+1} \times w_{s+1} \times c_{s+1}}$ as:

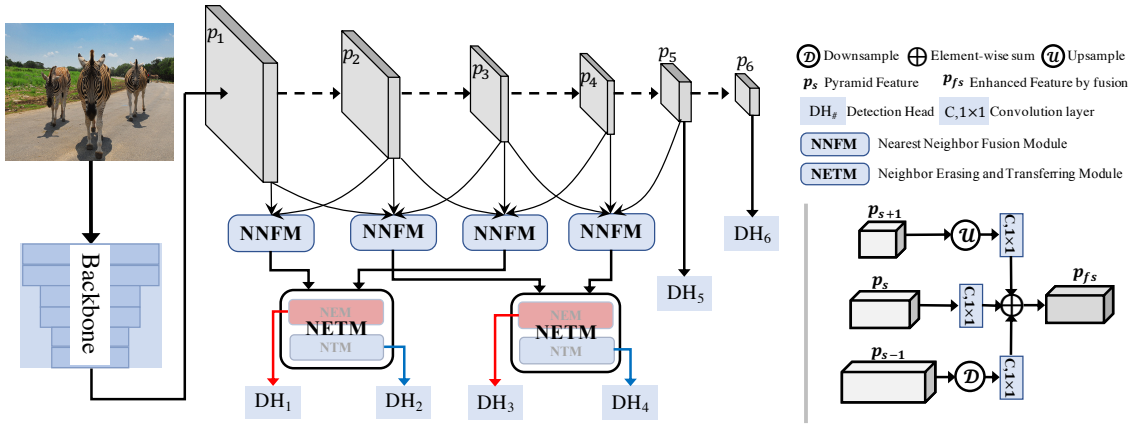
$$\begin{aligned} \tilde{p}_{s+1} &= \mathcal{T}_{s\to s+1}(p_{es}, p_{s+1}) \\ &= \mathcal{C}_{1\times1}(\mathcal{D}(p_{es}); W^{s+1}_s) \oplus p_{s+1}, \end{aligned} \quad (8)$$

composed of a downsampling operation $\mathcal{D}(.)$ to match the feature resolution and a convolutional layer $\mathcal{C}_{1\times1}$ with learnable $W^{s+1}_s \in \mathbb{R}^{1\times1\times c_s \times c_{s+1}}$ to maintain the consistent channel number. We perform an element-wise sum operation $\oplus$ to enhance p_{s+1} by combining the detailed information from p_{es}. We illustrate this Neighbor Transferring Module (NTM) in Fig. 3(c). The enhanced feature $\tilde{p}_{s+1}$ is used as the new pyramid feature for the subsequent scale-aware features generation and scale-aware object detection.

4. Single-Shot Detector: NETNet

Single-shot object detectors like SSD [35] directly carry out regression and classification based on predefined anchors. This provides the SSD with a better trade-off to achieve real-time detection and promising performance. However, SSD performs poorly for detecting small objects and also suffers from inaccurate localization, as shown in Fig. 1. To solve these problems, we design a new single-shot object detection network, called NETNet embedding the proposed NET mechanism as a scale-aware detector.

In NETNet, we build our backbone network as that of SSD. Taking the network with an input image size 300×300 as an example, we show the main network architecture of NETNet in Fig. 4(a). Features of six pyramid levels $\{p_1, p_2, p_3, p_4, p_5, p_6\}$ with resolutions $\{38\times38,\ 19\times19,\ 10\times10,\ 5\times5,\ 3\times3,\ 1\times1\}$ are extracted from the backbone as the basic feature pyramid. Based on the basic pyramid, we construct our NET Module (NETM) to generate scale-aware features and solve the aforementioned scale problems. In implementation, there are some scale-overlaps [34, 62] between the nearest neighbor pyramid lev-



(a) The main architecture of NETNet (b) Nearest Neighbor Fusion Module

Figure 4. The proposed NETNet. (a) The main architecture of NETNet. We illustrate this architecture with an input size of 300×300. Six pyramid layers are used for building detectors, as in SSD [35]. The embedded NNFM (b) is used for feature fusion before NETM.

els (*e.g.*, p_1 and p_2), when configuring the detection anchors and assigning ground truth. Therefore, we build the NETM in a skip manner. Additionally, considering that the scale-overlaps make features for one object existing in the nearest neighboring pyramid layers complementary, we introduce a Nearest Neighbor Fusion Module (NNFM) as shown in Fig. 4(b) to enhance the pyramid features firstly by fusing the nearest neighboring pyramid features. Based on the NNFM and NETM, six different detection heads for box regression and classification, are built upon the scale-aware features to construct our scale-aware detector NETNet. We present the details of NETM and NNFM as follows.

4.1. NETM in a Skip Manner

In typical single-shot detectors, features in the shallow layers (*e.g.*, p_1 with larger feature resolution 38×38) are used for detecting smaller objects, while features in deeper layers (*e.g.*, p_3 with smaller resolution 10×10) are used for detecting larger objects. Because features with small resolutions (*e.g.*, 3×3) have large receptive fields and less spatial information, we finally embed two NETMs in NETNet for feature erasing and transferring without using features p_5 and p_6. Due to the anchor configuration in SSD, two anchors in the nearest pyramid layers (*e.g.*, p_1 and p_2) may share the same ground truth. That is, one small object should be detected in p_1 and p_2 simultaneously. To avoid disturbing the overlapped supervision, our NETNet is elaborately designed by embedding two skipped NETMs.

One NETM is built upon the pyramid features of p_1 and p_3. To erase the features of larger objects from the shallow layer p_1, we first upsample p_3 and use a 1×1 convolution to generate soft spatial gate as Eq. 4 for larger objects. We evaluate the effects of several different spatial attention methods and choose channel-wise spatial attention as Eq. 6. Then, an erasing operation in Eq. 7 generates features for smaller objects. We also embed a light fusion module into NETM to make the generated scale-aware features more robust. The fusion module is constructed as a residual block as in [15] by stacking (1×1, 3×3, and 1×1 convolutional layers) with a skip connection. When applying the transferring module NTM, we first acquire the detailed information p_{es} that is helpful for larger objects from p_1 as Eq. 3. Then, this detailed information enhances the features p_3 as Eq. 8. The other NETM is built upon pyramid features of p_2 and p_4 with the similar configuration.

4.2. Nearest Neighbor Fusion Module

As pointed out in feature pyramid studies [19, 38], features from neighboring pyramid layers are complementary. Thus, incorporating context information from different layers promotes feature representation. Combining features from top to bottom is typically done to build a feature pyramid [10]. However, since our purpose is to remove large-object features from the shallow layers and generate scale-aware features, introducing other more scale features may increase the feature scale-confusion problem. Therefore, we propose a more effective fusion module, NNFM, to enhance the pyramid features.

As shown in Fig. 4(b), in NNFM, only features from the adjacent pyramid layers are fused as:

$$p_{fs} = \mathcal{H}_{s-1}(p_{s-1}) \oplus \mathcal{H}_s(p_s) \oplus \mathcal{H}_{s+1}(p_{s+1}), \quad (9)$$

where we denote the fused features of s^{th} pyramid layer as $p_{fs} \in \mathbb{R}^{h_s \times w_s \times c_s}$. $\mathcal{H}_{s-1}$ is constructed by a pooling layer and a 1×1 convolutional layer. $\mathcal{H}_s$ is constructed by a 1×1 convolutional layer. $\mathcal{H}_{s+1}$ is constructed by a bilinear upsampling layer and a 1×1 convolutional layer. Finally, these features are fused by an element-wise sum operation. Thus, we enhance the p_2 features by aggregating complementary information from p_1, p_2, and p_3, instead of using the features $\{p6, p5, p4, p3, p2\}$ like a top-down pyramid network. Performing NNFM will not aggravate the feature

Methods	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Baseline SSD	25.1	41.8	26.1	6.3	28.3	43.3
NEM	29.4	48.9	30.4	13.2	32.2	44.3
NTM	25.8	42.4	26.9	6.5	28.5	44.4
NETM	30.4	49.7	31.4	13.4	33.0	45.6
NETM + TDP	30.6	49.9	31.9	12.8	33.0	**46.3**
NETNet	**31.1**	**50.5**	**32.4**	**13.6**	**35.0**	45.4

Table 1. Evaluation for NETM and NNFM on COCO *minival*.

Methods	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Max Attention	28.7	47.3	29.9	11.5	31.4	43.4
Mean Attention	28.8	47.6	29.6	12.5	32.0	43.9
Global Attention	29.3	48.6	**30.5**	12.5	32.0	44.2
NEM	**29.4**	**48.9**	30.4	**13.2**	**32.2**	**44.3**

Table 2. Evaluation for different attention methods of NEM.

scale-confusion, since the information of tiny objects from p_1 is discarded using pooling operation and the information of larger objects from p_3 will be erased by the subsequent NEM. As a result, the features of objects which should be detected on p_2, are enhanced by fusing the complementary information with NNFM.

5. Experiments

Dataset: We evaluate our method on the benchmark detection dataset, MS COCO [32] dataset (*i.e.*, COCO). It has 80 object categories and more than 140k images. Following [35, 30], we train our NETNet on the union (*trainval35k*) of 80k training images and a 35k subset of validation images, and conduct ablation evaluations on the remaining 5k validation images (*minival*). The final results are obtained by testing on the 20k test images (*test-dev*) and submitted to the official server. The variations in scale of objects in COCO are complex. AP_s, AP_m, and AP_l evaluate the detection precision for three scales of objects.

Training protocols: We re-implement the SSD [35] as our baseline based on a Pytorch framework. All the models are trained over 160 epochs with the same training loss as SSD. For ablation experiments, we set the initial learning rate as 0.002 and decrease it by a factor of 0.1 after the 90^{th}, 120^{th}, and 140^{th} epochs, respectively. We follow [33], using a warm-up learning rate in the first 5 epochs. We set the weight decay to 0.0005 and the momentum to 0.9. Each model is trained with a batch size of 32 on 2 GPUs.

5.1. Ablation Study

Configuration of NETNet. For ablation experiments, we construct NETNet with a VGG-16 backbone pretrained on ImageNet [42], and train the models with an input size of 300×300. Following SSD, we truncate the final fully connected layers of the backbone and add a series of smaller convolutional layers to construct the feature pyramid.

Methods	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Baseline SSD	25.1	41.8	26.1	6.3	28.3	43.3
NEM_{13}	28.9	48.7	30.2	12.8	31.0	44.4
NEM_{24}	28.5	46.6	30.0	10.6	31.7	**44.5**
NNEM	29.1	48.8	30.1	12.7	31.9	44.4
NEM	**29.4**	**48.9**	**30.4**	**13.2**	**32.2**	44.3

Table 3. Evaluation for different configurations of NEM.

Evaluation of NETNet:

Overall NEM. As shown in Table 1, compared with SSD, NEM yields a large margin of absolute improvement of 4.3% AP. Because our NEM can remove the features of larger objects from the shallow layer to solve feature confusion, features for smaller objects can be activated to improve the performance for detecting smaller objects. We obtain a 6.9% AP improvement for small objects and 3.9% AP improvement for medium objects, which demonstrates the effectiveness of NEM for feature erasing.

NTM and NETM. We propose to transfer features using NTM to complement the detailed information of larger objects. As shown in Table 1, using only NTM brings a 1.1% improvement for large objects because of the enhanced features for large objects. Combining NEM and NTM promotes each module to learn better features through an adversarial strategy. Our NETM using NEM and NTM further improves the overall AP by 1.0%.

NNFM. We compare our NNFM with a typical Top-Down Pyramid (TDP) like FPN [30] based on our NETM. When combing the TDP with our NETM, a slight overall improvement, 0.2% AP, is achieved. However, we find the detection performance for small objects degrades by using TDP (from 13.4% AP to 12.8% AP), which may be caused by the feature confusion that is not consistent with our NET mechanism. When combining the NETM with NNFM (*i.e.*, NETNet), a 31.1% AP performance is obtained. Our NNFM further improves the performance for medium objects by a large margin (*i.e.*, 2.0%).

Evaluation of NEM:

Attention for NEM. We train our network with only two NEMs to evaluate different spatial gate generation methods as discussed in Sec. 3.2. Due to the large computation consumption of self-attention method in [53, 11], we only implement a simplified one as 'Global Attention' by reducing the inner channel number. As presented in Table 2, using attention as Eq. 6 in our NEM, which generates a channel-wise spatial gate for each channel of the shallow pyramid features, obtains a better performance of 29.4% AP.

NEM on different layers. We evaluate the influence of each NEM and show the results in Table 3. By only adding NEM on p_1 and p_3 as NEM_{13}, we obtain a 6.5% AP improvement on AP_s, which is better than that of NEM_{24} (on p_2 and p_4) because there are more small objects features in p_1. We obtain a better improvement for medium objects by

Methods	Backbone	Image Size	Time (ms)	FPS	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Anchor-free detectors:										
CornerNet [27]	Hourglass-104	511×511	244	4.1	40.5	56.5	43.1	19.4	42.7	53.9
CenterNet [8]	Hourglass-104	511×511	340	2.9	44.9	62.4	48.1	25.6	47.4	57.4
FCOS [47]	Res101-FPN	1333×800	-	-	41.5	60.7	45.0	24.4	44.8	51.6
Single-stage detectors:										
SSD300 [35]	VGG-16	300×300	17*	58.9	25.1	43.1	25.8	6.6	25.9	41.4
DFPR [22]	VGG-16	300×300	-	-	28.4	48.2	29.1	-	-	-
PFPNet-S300 [19]	VGG-16	300×300	-	-	29.6	49.6	31.1	10.6	32.0	44.9
RefineDet320 [56]	VGG-16	320×320	26	38.7	29.4	49.2	31.3	10.0	32.0	44.4
RFBNet [33]	VGG-16	300×300	15 (19*)	66.7	30.3	49.3	31.8	11.8	31.9	45.9
EFIP [38]	VGG-16	300×300	14	71.4	30.0	48.8	31.7	10.9	32.8	46.3
HSD [2]	VGG-16	320×320	25	40.0	33.5	53.2	36.1	15.0	35.0	47.8
NETNet (ours)	VGG-16	300×300	18	55.6	32.0	51.5	33.6	13.9	34.5	46.2
NETNet+Ref [2]	VGG-16	320×320	-	-	34.9	53.8	37.8	16.3	37.7	48.2
DSSD513 [10]	ResNet-101	513×513	182	5.5	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [31]	ResNet-101	500×500	90	11.1	34.4	53.1	36.8	14.7	38.5	48.5
STDN512 [61]	DenseNet-169	513×513	-	-	31.8	51.0	33.6	14.4	36.1	43.4
DFPR [22]	ResNet-101	512×512	-	-	34.6	54.3	37.3	14.7	38.1	51.9
RefineDet512 [56]	ResNet-101	512×512	-	-	36.4	57.5	39.5	16.6	39.9	51.4
SSD512 [35]	VGG-16	512×512	28	35.7	28.8	48.5	30.3	10.9	31.8	43.5
DES512 [59]	VGG-16	512×512	-	-	32.8	53.2	34.6	13.9	36.0	47.6
RFBNet [33]	VGG-16	512×512	33 (37*)	30.3	34.4	55.7	36.4	17.6	37.0	47.6
EFIP [38]	VGG-16	512×512	29	34.5	34.6	55.8	36.8	18.3	38.2	47.1
TripleNet [3]	ResNet-101	512×512	-	-	37.4	59.3	39.6	18.5	39.0	52.7
NETNet (ours)	VGG-16	512×512	33	30.3	36.7	57.4	39.2	20.2	39.2	49.0
NETNet (ours)	ResNet-101	512×512	37	27.0	38.5	58.6	41.3	19.0	42.3	53.9

Table 4. Comparison on the COCO *test-dev* set. The results are reported for the case of single-scale inference. We test the time on a Titan X Pascal GPU with Pytorch 0.3.1. Times with * are obtained by testing in the same environment with NETNet.

NEM_{24}. There is some ground truth and feature overlap in p_1 and p_2, which yields the improvements for both small and medium objects using each NEM. We obtain the best result by combining them. These results demonstrate the effectiveness of our method for erasing redundant features.

Skipped NEM. We also construct a model by adding three regular NEMs built upon (p_1, p_2), (p_2, p_3), and (p_3, p_4), respectively. This is a type of nearest neighbor erasing module built upon the features of two nearest neighbor layers. We denote this model as NNEM in Table 3. The NNEM model obtains a lower performance (29.1%) than our NEM (29.4%). Because the same ground truth may be assigned to predefined anchors from two neighboring layers, using NNEM disturbs the ground truth supervision. Using the skipped NEM eases the network training for better results.

Evaluation of network configurations: We evaluate the performance of NETNet with different configurations. By refining the learning rate (using 0.004 as the initial learning rate), we achieve a final best performance of 31.8% AP with a 300×300 input size. When we further use the refined prediction procedure in [2], a 34.7% AP performance is obtained. In addition, larger image size and better backbone help improve the performance. With VGG-16 and a 512×512 size, 36.1% AP is obtained. Using ResNet-101 brings NETNet to a top performance, 38.2% AP.

5.2. Results on COCO Test Set

We evaluate NETNet on the COCO *test-dev* set and compare it with previous state-of-the-art methods, as shown in Table 4. Our NETNet outperforms the baseline SSD significantly with only a slight extra time cost. With an input size of 300×300 and VGG-16, our NETNet obtains 32.0% AP with 55.6 FPS, which outperforms other state-of-the-art single-shot detectors with a similar configuration. Employing the refinement in [2] helps NETNet obtain a top performance 34.9% AP. When testing with an image size of 512×512, NETNet obtains 36.7% (30.3 FPS) with VGG-16 and 38.5% (27.0 FPS) with ResNet-101. Some anchor-free methods achieve better detection accuracy, but they are generally require more than 100 ms to process one image. As shown in Fig. 5, our method achieves an better trade-off for accurate detection while maintaining a fast speed.

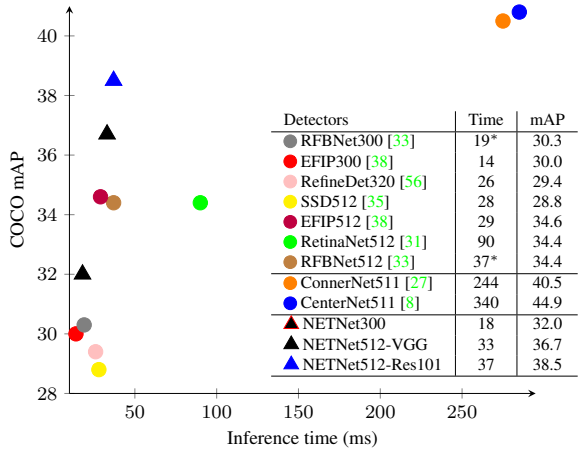


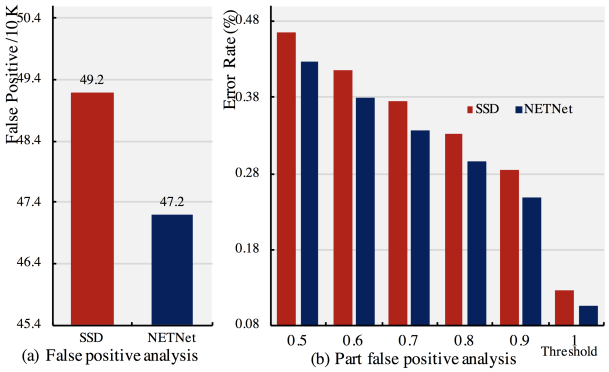
Figure 5. Accuracy (mAP) vs. speed (ms) comparison. Methods in the top-left corner have better overall performance.

6. Discussion

Different from previous pyramid methods, NET mechanism helps reconfigure the basic pyramid to be scale-aware features which are more suitable for scale-aware detection. In another side, because we need use shallow features to generate deep features by progressively convolution opera-

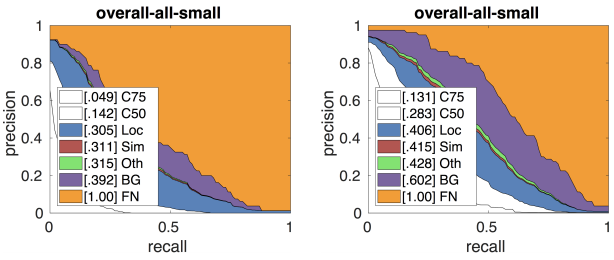
(a) baseline (b) ours (c) baseline (d) ours

Figure 6. Detection results visualization. Our method can alleviate the part false positive problem as (b) and the small objects missing (false negative problem) as (d).



(a) False positive analysis (b) Part false positive analysis

Figure 7. Error analysis for total false positive problem (a) and part false positive problem (b) on COCO *minival*.



(a) SSD (b) NETNet

Figure 8. Error analysis for (a) baseline SSD and (b) our NETNet on small objects. 'FN' represents the missing detection error (false negative). The overall false negative error can be measured by subtracting the AP value of BG from FN. The overall false negative error is 60.8% for SSD and 39.8% for NETNet. Lower is better.

tions in a network, using a direct hard supervision will force the large object regions in shallow layers of the backbone to be background and harm the feature learning of deep layers. NET works like a soft supervision by introducing a reversed feedback from high-level features for feature erasing, which will not harm the feature learning but enhance the information aggregation in the backbone pyramid.

In addition, we carry out an error analysis to further demonstrate the effectiveness of our method for solving the false positive (FP) problem and false negative problem (FN, *i.e.*, missing detection). For comparison, we use the detection results on the *minival* set by SSD and NETNet (31.8% AP) with VGG-16 and 300×300 image size.

Tackling FP problem. By treating the predicted box, which has a IoU <0.5 with the ground truth as a FP sample, we conduct a statistical analysis for the FP problem. In total, there are about 20k less FP samples by our method than SSD as shown in Fig. 7(a), which demonstrates our method can alleviate this problem. We further analyze the part false positive (PFP) problem based on the PFP samples under different thresholds τ. The part rate p_θ is calculated as the ratio of intersection region (between one predicted FP box and the ground truth) over the area of the predicted box. If $p_\theta > \tau$, the FP box is regarded as a PFP sample. As shown in Fig. 7(b), the x-axis denotes the thresholds and y-axis represents the ratio of PFP sample number over total predicted box number. Our method can reduce the PFP error. We visualize some results in Fig. 6 (a) and (b).

Tackling FN problem. We show the error analysis plots of our baseline SSD and our NETNet in Fig. 8 for small objects. Each plot describes a Precision Recall (PR) curve obtained by eliminating the corresponding detection errors except 'C75' (*i.e.*, AP_{75}) and 'C50'(*i.e.*, AP_{50}). Thus, the area of each color can measure the corresponding errors. Overall, our method is more significant on small object detection (*i.e.*, 39.8% FN error by NETNet *vs* 60.8% error by SSD). As shown in Fig. 6(d), our NETNet can detect small objects precisely, and alleviate the FN problem well.

7. Conclusion

In this paper, we have proposed a Neighbor Erasing and Transferring (NET) mechanism with feature reconfiguration for tackling complex scale variations in object detection. Scale-aware features are generated by erasing the features of larger objects from the shallow layers and transferring them into deep pyramid layers. We have constructed a single-shot network called NETNet by embedding NETM and NNFM to achieve fast and accurate scale-aware object detection. As demonstrated by experiments on the MS COCO dataset, our NETNet is able to solve the missing detection and part false positive problems effectively, leading to an improved trade-off for real-time and accurate detection. In future work, we consider to explore the advantages of NET on other detectors for scale-aware object detection.

Acknowledgement This work was supported by the National Key R&D Program of China (Grant Nos. 2018AAA0102800 and 2018AAA0102802), National Natural Science Foundation of China (Grant No. 61632018, Grant Nos. 61906131), Postdoctoral Program for Innovative Talents (No. BX20180214), China Postdoctoral Science Foundation (No. 2018M641647).

References

[1] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: delving into high quality object detection. In *CVPR*, 2018. 2

[2] Jiale Cao, Yanwei Pang, Jungong Han, and Xuelong Li. Hierarchical shot detector. In *ICCV*, 2019. 7

[3] Jiale Cao, Yanwei Pang, and Xuelong Li. Triply supervised decoder networks for joint detection and segmentation. In *CVPR*, 2019. 7

[4] Shuhan Chen, Xiuli Tan, Ben Wang, and Xuelong Hu. Reverse attention for salient object detection. In *ECCV*, 2018. 3

[5] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-FCN: object detection via region-based fully convolutional networks. In *NeurIPS*, 2016. 1, 2

[6] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017. 2

[7] Xingping Dong and Jianbing Shen. Triplet loss in siamese network for object tracking. In *ECCV*, pages 472–488, 2018. 1

[8] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *ICCV*, 2019. 7

[9] Deng-Ping Fan, Wenguan Wang, Ming-Ming Cheng, and Jianbing Shen. Shifting more attention to video salient object detection. In *CVPR*, 2019. 3

[10] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C. Berg. DSSD : Deconvolutional single shot detector. *CoRR*, abs/1701.06659, 2017. 2, 3, 5, 7

[11] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *CVPR*, 2019. 3, 4, 6

[12] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V. Le. NAS-FPN: learning scalable feature pyramid architecture for object detection. In *CVPR*, 2019. 2

[13] Ross B. Girshick. Fast R-CNN. In *ICCV*, 2015. 1, 2

[14] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5

[16] Qibin Hou, Peng-Tao Jiang, Yunchao Wei, and Ming-Ming Cheng. Self-erasing network for integral object attention. In *NeurIPS*, 2018. 3

[17] Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. Acquisition of localization confidence for accurate object detection. In *ECCV*, 2018. 2

[18] Nie Jing, Anwer Rao Muhammad, Cholakkal Hisham, Khan Fahad Shahbaz, Pang Yanwei, and Shao Ling. Enriched feature guided refinement network for object detection. In *ICCV*, 2019. 1

[19] Seung-Wook Kim, Hyong-Keun Kook, Jee-Young Sun, Mun-Cheon Kang, and Sung-Jea Ko. Parallel feature pyramid network for object detection. In *ECCV*, 2018. 2, 5, 7

[20] Yonghyun Kim, Bong-Nam Kang, and Daijin Kim. SAN: learning relationship between convolutional features for multi-scale object detection. In *ECCV*, 2018. 1

[21] Alexander Kirillov, Ross B. Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, 2019. 2

[22] Tao Kong, Fuchun Sun, Wen-bing Huang, and Huaping Liu. Deep feature pyramid reconfiguration for object detection. In *ECCV*, 2018. 2, 7

[23] Tao Kong, Fuchun Sun, Huaping Liu, Yuning Jiang, and Jianbo Shi. Consistent optimization for single-shot object detection. *CoRR*, abs/1901.06563, 2019. 2

[24] Tao Kong, Fuchun Sun, Anbang Yao, Huaping Liu, Ming Lu, and Yurong Chen. RON: reverse connection with objectness prior networks for object detection. In *CVPR*, 2017. 1, 2

[25] Tao Kong, Anbang Yao, Yurong Chen, and Fuchun Sun. Hypernet: Towards accurate region proposal generation and joint object detection. In *CVPR*, 2016. 2

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 1

[27] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *ECCV*, 2018. 1, 7

[28] Yanghao Li, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Scale-aware trident networks for object detection. In *ICCV*, 2019. 2

[29] Zeming Li, Chao Peng, Gang Yu, Xiangyu Zhang, Yangdong Deng, and Jian Sun. Light-head R-CNN: in defense of two-stage object detector. *CoRR*, abs/1711.07264, 2017. 2

[30] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 2, 3, 6

[31] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 1, 2, 7

[32] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, 2014. 6

[33] Songtao Liu, Di Huang, and Yunhong Wang. Receptive field block net for accurate and fast object detection. In *ECCV*, 2018. 2, 6, 7

[34] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *CVPR*, 2018. 4

[35] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. In *ECCV*, 2016. 1, 2, 3, 4, 5, 6, 7

[36] Shuai Ma, Yanwei Pang, Jing Pan, and Ling Shao. Preserving details in semantics-aware context for scene parsing. *Sci. China Inf. Sci.*, 2020. 1

[37] Yanwei Pang, Yazhao Li, Jianbing Shen, and Ling Shao. Towards bridging semantic gap to improve semantic segmentation. In *ICCV*, 2019. 2

[38] Yanwei Pang, Tiancai Wang, Rao Muhammad Anwer, Fahad Shahbaz Khan, and Ling Shao. Efficient featurized im-

age pyramid network for single shot detector. In *CVPR*, 2019. 3, 5, 7

[39] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 1, 2

[40] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018. 1

[41] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 2

[42] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 6

[43] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 1

[44] Bharat Singh and Larry S. Davis. An analysis of scale invariance in object detection SNIP. In *CVPR*, 2018. 1

[45] Bharat Singh, Mahyar Najibi, and Larry S. Davis. SNIPER: efficient multi-scale training. In *NeurIPS*, 2018. 1

[46] Hongmei Song, Wenguan Wang, Sanyuan Zhao, Jianbing Shen, and Kin-Man Lam. Pyramid dilated deeper convlstm for video salient object detection. In *ECCV*, 2018. 2

[47] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, 2019. 7

[48] Wenguan Wang, Qiuxia Lai, Huazhu Fu, Jianbing Shen, and Haibin Ling. Salient object detection in the deep learning era: An in-depth survey. *CoRR*, abs/1904.09146, 2019. 2

[49] Wenguan Wang and Jianbing Shen. Deep visual attention prediction. *TIP*, 2018. 3

[50] Wenguan Wang, Jianbing Shen, Ming-Ming Cheng, and Ling Shao. An iterative and cooperative top-down and bottom-up inference network for salient object detection. In *CVPR*, 2019. 1

[51] Wenguan Wang, Jianbing Shen, Xingping Dong, and Ali Borji. Salient object detection driven by fixation prediction. In *CVPR*, 2018. 2

[52] Wenguan Wang, Shuyang Zhao, Jianbing Shen, Steven C. H. Hoi, and Ali Borji. Salient object detection with pyramid attention and salient edges. In *CVPR*, 2019. 3

[53] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 3, 4, 6

[54] Yunchao Wei, Jiashi Feng, Xiaodan Liang, Ming-Ming Cheng, Yao Zhao, and Shuicheng Yan. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *CVPR*, 2017. 3

[55] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. CBAM: convolutional block attention module. In *ECCV*, 2018. 4

[56] Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z. Li. Single-shot refinement neural network for object detection. In *CVPR*, 2018. 7

[57] Xiaolin Zhang, Yunchao Wei, Jiashi Feng, Yi Yang, and Thomas S. Huang. Adversarial complementary learning for weakly supervised object localization. In *CVPR*, 2018. 3

[58] Zhijie Zhang and Yanwei Pang. Cgnet: cross-guidance network for semantic segmentation. *Sci. China Inf. Sci.*, 2020. 1

[59] Zhishuai Zhang, Siyuan Qiao, Cihang Xie, Wei Shen, Bo Wang, and Alan L. Yuille. Single-shot object detection with enriched semantics. In *CVPR*, 2018. 1, 2, 7

[60] Qijie Zhao, Tao Sheng, Yongtao Wang, Zhi Tang, Ying Chen, Ling Cai, and Haibin Ling. M2det: A single-shot object detector based on multi-level feature pyramid network. In *AAAI*, 2019. 2

[61] Peng Zhou, Bingbing Ni, Cong Geng, Jianguo Hu, and Yi Xu. Scale-transferrable object detection. In *CVPR*, 2018. 2, 7

[62] Yousong Zhu, Chaoyang Zhao, Jinqiao Wang, Xu Zhao, Yi Wu, and Hanqing Lu. Couplenet: Coupling global structure with local parts for object detection. In *ICCV*, 2017. 4