ASAN: Self-Attending and Semantic Activating Network towards Better Object Detection

Xinyu ZHU†, Jun ZHANG†, Nonmembers, and Gengsheng CHEN†, Member

SUMMARY
Recent top-performing object detectors usually depend on a two-stage approach, which benefits from its region proposal and refining practice but suffers low detection speed. By contrast, one-stage approaches have the advantage of high efficiency while sacrifice their accuracies to some extent. In this paper, we propose a novel single-shot object detection network which inherits the merits of both. Motivated by the idea of semantic enrichment to the convolutional features within a typical deep detector, we propose two novel modules: 1) by modeling the semantic interactions between channels and the long-range dependencies between spatial positions, the self-attending module generates both channel and position attention, and enhance the original convolutional features in a self-guided manner; 2) leveraging the class-discriminative localization ability of classification-trained CNN, the semantic activating module learns a semantic meaningful convolutional response which augments low-level convolutional features with strong class-specific semantic information. The so called self-attending and semantic activating network (ASAN) achieves better accuracy than two-stage methods and is able to fulfill real-time processing. Comprehensive experiments on PASCAL VOC indicates that ASAN achieves state-of-the-art detection performance with high efficiency.

key words: object detection, self-attending, semantic activating

1. Introduction

Object detection, aiming at locating object instances in natural images that belong to a group of pre-defined categories, is one of the most challenging and fundamental problems in computer vision and has attracted much research interest.

In recent years, the significant advances of deep learning techniques have led to remarkable breakthroughs in the field of object detection. The current deep neural networks (DNNs) based detectors can be briefly divided into two categories: 1) the two-stage approaches, represented by Faster R-CNN [1] and R-FCN [2], include the first step of generating category-agnostic region proposals, followed by a classification and regression step according to per-proposal CNN based deep features. Benefiting from the two-stage structure and the two-stage discriminative representations, these approaches have been reaching top performances on several challenging benchmarks, including PASCAL VOC [3] and MS COCO [4]; 2) the one-stage approaches, represented by YOLO [5] and SSD [6], directly predict class probabilities and bounding box offsets from the whole image by dense sampling of different positions, scales and ratios. Since all computation is encapsulated in a single feed-forward CNN network through a unified pipeline, the one-stage approaches are elegant and highly efficient, but tend to sacrifice accuracies.

It is generally accepted that, in both kinds of approaches, CNN architecture serves as the core engine in a detector and CNN representations are of primary importance to the overall detection performance. A great deal of effort has been made aiming at generating better feature representations, where a typical strategy is to apply stronger CNN backbones like ResNet [7] and Inception [8] for feature extraction as in researches [7], [9] and [10].

Another popular strategy is to leverage the inherent multi-scale pyramid structure of CNN, integrating low-level geometric information and high-level semantic information in different forms. Approaches of multi-scale detection broadly fall into three types. Figure 1 (a)–(c) illustrates their structures. The first type combines predictions from multiple feature maps to handle objects of various scales. Its representative approaches include MSCNN [11] and SSD [6]. Such approaches explore the effect of different receptive field sizes in different convolutional layers and learn semantic information in a hierarchical manner. However, the problem is that smaller objects might not be well detected from the low-level features with rich geometric details but little semantic information. The second type, represented by ION [12] and HyperNet [13], generates predictions according to the combined features of multiple layers, which are more descriptive but computational expensive due to the high dimensionality. The third type is a combination of the above two, which appends a top-down feedback network parallel to the standard bottom-up network to transmit back the high-level semantic features to lower layers. Predictions are then made at multiple layers based on the integrated features. Its representative methods include FPN [14], DSSD [10], RON [15], STDN [16] and RefineDet [17]. The appended reverse fusion network enables these approaches to effectively alleviate the drawbacks of poor semantics in lower layers or high feature dimensionality. However, it still leaves behind several problems: the reverse fusion units used for feature selection will bring about extra computational cost and the reverse connections slow down the overall processing speed of the detectors.

Motivated by the problems discussed above, in this paper, we develop a novel single-shot detection network by simultaneously taking efficient feature enhancement and semantic enrichment into consideration, named self-.
Attending and Semantic Activating Network (ASAN).

We propose a simpler method to enhance the original CNN feature representations in a one-stage multi-scale detector without involving reverse connections (Fig. 1 (d)), promising them with high robustness to spatial translation and discriminative semantic encoding ability. The architecture of ASAN is shown in Fig. 2, where two major components, the self-attending module and the semantic activating module are brought in.

Self-Attending Module: Previous works [12], [18] have provided strong evidence that capturing semantic interactions between objects in a wider range or at the global level can effectively boost the performance of a detector. In our self-attending module, we adopt attention mechanism to learn global feature dependencies in a self-guided manner in both channel and spatial dimensions, and selectively emphasize or depress the original features. Specifically, we insert channel attention to model the semantic interactions between channels, telling the network ‘what to focus on’, and insert position attention to model the long-range dependencies between spatial positions, telling the network ‘where to look’.

Semantic Activating Module: As discovered in previous research [19], convolutional units in deep CNNs have the ability to localize discriminative semantic parts with no spatial supervision provided. Thus, in our semantic activat-
ing module, we train an extra multi-label classification task. With an implicit use of the class activation mapping (CAM) mechanism [20], we generate a class-specific and semantic meaningful convolutional response, and apply it to activate the low-level features using strong high-level semantic information. The activated features are then propagated forward and further enhanced by the self-attending module, producing more powerful feature representations for detection.

We append these two modules to a typical one-stage and multi-scale detection framework, e.g. SSD [6], and train the whole network in an end-to-end manner. In ASAN, semantic booming starts from lower layers, and progressively influences and meanwhile gets enhanced by the upper layers. This feed-forward architecture enables our ASAN to be both fast and accurate.

Our major contributions are as follows:

- We propose to enhance the feature representations of deep learning based object detector with two novel modules, the self-attending module that captures the global semantic dependencies in both channel and spatial dimensions, and the semantic activating module that augments low-level features with class-specific semantic information.

- We apply the two modules to a one-stage multi-scale detection framework to form a simple feed-forward architecture. The newly proposed detector shows a significant performance gain with budgeted computation addition.

- We conduct extensive experiments on two published benchmarks (PASCAL VOC and MS COCO). Experimental results show that ASAN achieves a superior performance in precision at a real time processing speed.

2. Related Work

2.1 Object Detection

Object detection has gone through a rapid evolution since the successful application of DCNNs by R-CNN [21]. R-CNN formulates detection as a two-stage problem with a dedicated pre-processing stage for region proposal. Its advanced variations, including Fast R-CNN [22], Faster R-CNN [1] and Mask R-CNN [23], keep on promoting the representational power of the original convolutional features and have narrowed this performance gap, achieving comparable or even better precision scores than many two-stage approaches while retaining high efficiency.

Actually, most of the newly proposed deep learning based detectors are built on one of the above mentioned frameworks, improving them in different dimensions. Feasible improvements include producing enhanced object representations by exploiting multi-layer features [12], [13], [15]–[18] as we have discussed in Sect. 1, and context modeling by exploring global [12], [26] or local [27], [28] contextual information. In this paper, we enhance representations by an extra semantic activating practice and capture global semantic context via the self-attending mechanism.

2.2 Self-Attention Modules

Attention mechanism has been widely investigated in natural language processing (NLP), usually coordinated with recurrent neural networks (RNNs), to model long-range feature dependencies. The concept of self-attention is initially raised by research [29], in which attention is generated purely based on the internal correlation of features before being applied to the machine translation task.

Since self-attention can be easily integrated in CNNs and can lead to more interpretable and powerful representations, it is increasingly applied in computer vision. Research [30], although not explicitly stated in the paper, introduces self-attention mechanism to image classification task, learning 3D attention residual within a stacked hourglass network. By modeling the inter-dependencies with channel-wise attention, research [31] proposes a squeeze-and-excitation block to recalibrate feature responses between channels for better classification. Research [32] proposes a position-wise self-attention based relation module for object detection, which models the spatial relationships between objects and generates geometric weights to help improve recognition and localization.

More recently, researches [33]–[35] study the combined use of channel and position attention in different visual tasks such as image classification, object detection and scene segmentation. In this paper, we propose an adapted self-attending module learning both channel and position attention for object detection. It effectively enhances the representational power of the original convolutional features and is well-designed to be embedded in typical multi-scale detection frameworks.

2.3 Semantic Enrichment

In addition to the multi-scale detectors that leverage the inherent semantic representations in deep CNNs, a lot of approaches have been proposed to explicitly enhance feature semantics. Several recent researches [23], [26] have proved that the semantic segmentation aware CNN features trained under the pixel-wise supervision can facilitate object detection considering the close connection between segmentation and detection. Research [26] trains semantic segmentation
for contextually priming region proposal and providing iterative feedback to enhance the entire network. Research [23] trains an instance segmentation branch in parallel to the original object detection branch, and yields prominent detection performance gain.

These methods can help improve detection benefiting from the more elaborate semantic segmentation features. However, the required manually-labeled segmentation annotations are extremely cost-prohibitive. To counter this problem, researches [27] and [36] propose to learn activation maps in a weakly supervised manner, using only the provided bounding box annotations for detection. Concretely, research [27] concatenates detection features and segmentation features at the highest level as the final object representation, while research [36] adopts a similar semantic enriching implementation as ours to generate segmentation-aware response at the lowest detection layer and take it as the semantic activating signal.

In this paper, we propose to enrich semantics in a much more simple way. We train a lightweight multi-label classification branch instead of the segmentation branch to take full advantage of the intrinsic convolutional responses to discriminative semantic parts.

3. Proposed Method

3.1 Overview

As shown in Fig. 2, our ASAN mainly contains three parts: a one-stage multi-scale detection network, a self-attending module to generate semantic intensive features at each detection source layer, and a semantic activating module to achieve low-level semantic augmentation explicitly.

We use the same network from SSD [6] as our backbone detection network. In brief, it uses a VGG16 [37] architecture, where its fc6 and fc7 layers are converted to convolutional layers, appended by a series of extra convolutional layers (from conv6_1 to conv9_2 for SSD300 and from conv6_1 to conv10_2 for SSD512). Predictions are generated from multiple feature maps with different spatial resolutions and receptive fields.

Motivated by the idea of enhancing feature semantics using no extra reverse fusion network, we bring in self-attention mechanism to produce semantic intensive features. As shown in the lower part of Fig. 2, the proposed self-attending module contains several self-attending blocks, each attached to a detection source layer. During training, these blocks initially capture long-range semantic dependencies of local features and learn weighted attentions in both channel and spatial dimensions. We utilize the learned attentions to selectively emphasize or depress the original features. The resulting enhanced features are then used for producing predictions and meanwhile used as inputs to subsequent layers.

As discussed in Sect. 1, to deal with the problem of poor semantic representation at lower layers, we employ class activation mapping mechanism to explicitly provide with rich semantic information in a weakly-supervised manner. The semantic activating module contains a multi-label classification branch. Using only the object label annotation, it generates class-specific and semantic meaningful convolutional response, which is used to activate the original low-level features. We apply this module to the lowest detection layer (conv4_3) in front of all the self-attending blocks so that the activated semantic-aware features can be propagated forward and get further enhanced throughout the network.

3.2 Self-Attending Module

In our self-attending module, two types of attentions, the channel attention and the position attention, are used to draw global dependencies over local features, with its details shown in Fig. 3.

**Channel Attention.** In deep CNNs, each filter is responsible for detecting specific visual patterns, and different filter responses are associated with each other according to their semantic dependencies. There also exist noisy filters that respond to meaningless patterns and bring in redundancies to the feature maps. In order to emphasize discriminative filters as well as to exploit their inter-dependencies, and at the same time to depress the invalid responses, we introduce a channel attention branch to model the inter-channel relationships at a global level.

The detailed architecture of our channel attention branch is illustrated in the upper part of Fig. 3. First, we aggregate the features in each channel with a global average pooling layer (abbr. GAP) to eliminate the spatial information. Denoting the input features as \( X \in \mathbb{R}^{C \times H \times W} \), we get the channel matrix \( F_k^{ch} \in \mathbb{R}^{C \times 1 \times 1} \) by:

\[
F_k^{ch} = GAP(X_k) = \frac{1}{H \cdot W} \sum_{j,l} X_{kj,li}.
\]

Next, we encode the global channel information by a multi-layer perceptron with one hidden layer, and apply a sigmoid function to obtain the attention weights. We compute the channel attention matrix \( A^{ch} \in \mathbb{R}^{C \times 1 \times 1} \) as:

\[
A^{ch} = Attention^{ch}(X) = \sigma(W_2 \cdot ReLU(W_1 \cdot GAP(X) + b_1) + b_2),
\]

where \( W_1 \in \mathbb{R}^{C \times C}, b_1 \in \mathbb{R}^C, W_2 \in \mathbb{R}^{C \times C'}, b_2 \in \mathbb{R}^C \). To avoid high computational cost, we set the reduction ratio of the hidden layer \( r = C/C' \) to be 8 in all our experiments (which will be discussed in Sect. 4.4).

Finally, we broadcast \( A^{ch} \) and perform element-wise multiplication with the input feature \( X \) to achieve the final output \( X_{kj}^{ch} \in \mathbb{R}^{C \times H \times W} \) as follows:

\[
X_{kj}^{ch} = A_k^{ch} \cdot X_{kj,li}.
\]

It can be inferred from Eq. (3) that the values of the generated channel attention \( A^{ch} \) determines whether a filter should be highly weighted in the resulting features.
**Position Attention.** Semantic contextual information and object relationships within a single scene are of great importance for accurate visual recognition and localization. To capture such relationships and to tell distinctive spatial locations that the network should focus on apart from other background regions, we introduce a position attention branch that exploits inter-position dependencies and encodes semantic contextual information into local features from a wider range.

The lower part in Fig. 3 presents the detailed architecture of the position attention branch. Given the input feature $X \in R^{C \times H \times W}$, we first aggregate spatial information of the feature maps using a global average pooling layer along the channel axis, and get $F_{po} \in R^{1 \times H \times W}$ by:

$$F_{po}^{ji} = GAP^{\text{channel}}(X^{ji}) = \frac{1}{C} \sum_{k} X^{kj \ i}.$$ \hspace{1cm} (4)

With informative regions highlighted in $F_{po}$ after the pooling operation, we directly apply a convolutional layer with output dimension of 1, followed with a sigmoid function to generate the position attention matrix $A_{po} \in R^{1 \times H \times W}$:

$$A_{po} = Attention_{po}(X) = \sigma(Conv^{n \times n}_{po}(GAP^{\text{channel}}(X))),$$ \hspace{1cm} (5)

where $Conv^{n \times n}_{po}$ denotes convolutional layer with kernel size of $n \times n$ configured with the activation operation.

In order to model long-range spatial dependencies, we make a brute use of large convolutional filters. Large kernels that enable dense connections between feature maps have already been proved in research [38] to be effective on handling different transformations and enlarging valid receptive fields without loss of information.

Considering the different resolutions and receptive fields of the original feature maps, we carefully choose kernel sizes that accommodate to different detection source layers. Specifically, for lower detection layers in SSD300, e.g. conv4_3 and fc7, which have relatively higher resolutions and smaller receptive fields, we set the kernel size to be $9 \times 9$. For detection layers conv6_2 and conv7_2, we assign $7 \times 7$ and $5 \times 5$ convolutions respectively. For deeper layer conv8_2, we further reduce the kernel size to $3 \times 3$. Since feature maps from conv9_2 in SSD300 already have the size of $1 \times 1$, we ignore the position attention branch and keep only the channel attention branch for this layer.

Analogous to the channel attention branch, we tile $A_{po}$ along channel axis and perform element-wise multiplication with the input feature $X$ to achieve $X^{po} \in R^{C \times H \times W}$ as follows:

$$X^{po}_{kj \ i} = A_{po}^{ji} \cdot X^{kj \ i}.$$ \hspace{1cm} (6)

**Combination.** To get the final semantic intensive features, we simply apply an element-wise summation to the outputs from the two attention branches:

$$X^{\text{Intensive}} = X^{ch} \oplus X^{po}.$$ \hspace{1cm} (7)

$X^{\text{Intensive}} \in R^{C \times H \times W}$ is encoded with global-level channel-wise interaction as well as long-range position-wise dependencies, which is used to replace the original $X$ in the forward pass.

3.3 Semantic Activating Module

In our semantic activating module, semantic information is exploited in a multi-label classification branch using a modification of the class activation mapping (CAM) [20] technique.

Figure 4 (a) gives a simple illustration of the CAM technique. CAM uses global average pooling as a structural regularizer and computes a weighted sum of the global responses to generate probability scores. The learned weights directly indicate the importance of each filter leading to the classification of an image to a particular category. Since deep filters reacts to the presence of specific visual patterns at different spatial locations, the CAM technique allows the
classification-trained convolutional network to both classify the image and localize class-specific image regions.

We consider this kind of ability naturally meets our need for semantic augmentation, which identifies and localizes discriminative semantic regions relevant to particular categories. Thus, we take an implicit use of the CAM mechanism in our semantic activating module, as shown in Fig. 4 (b). We generate deep representation $F \in \mathbb{R}^{C \times H \times W}$ from low-level convolutional feature $X \in \mathbb{R}^{C \times H \times W}$ by attaching several extra layers to the low detection layer (conv4_3), which can be represented by:

$$F = \text{Conv}^{\text{extra}}(X),$$

where $\text{Conv}^{\text{extra}}$ denotes the operation of the attached layers.

In analogy with the standard class activation mapping procedure, we apply global average pooling to $F$ and map the resulting matrix into class scores $S \in \mathbb{R}^{C}$:

$$S = \text{softmax}(W \cdot \text{GAP}(F) + b),$$

where $W \in \mathbb{R}^{C \times C}$, $b \in \mathbb{R}^{C}$, and $C$ is the number of pre-defined categories.

We compare the output scores with the ground-truth vector $G \in \mathbb{R}^{C}$, which is given by:

$$[\mathbb{I}(1), \ldots, \mathbb{I}(i), \ldots, \mathbb{I}(c)],$$

where $\mathbb{I}(i)$ equals value 1 if there exist objects in the image that belong to the $i^{th}$ category. This ground-truth vector can be directly inferred from detection supervision with no extra annotation required.

Optimized by such multi-label classification supervision, $F$ is fully encoded with semantic knowledge, with each channel activated by a certain semantic part of object. Instead of computing the weighted sum of the feature maps to get the class activation map, we require the network to further integrate the deep representation $F$ and learn a convolutional response $A$ with the same size of the original feature $X$, where

$$A = \text{Activation}(F) = \text{Conv}^{1 \times 1}(F).$$

Here, we simply employ a $1 \times 1$ convolution that performs channel correlation learning at each position.

Finally, semantic activation is achieved by applying $A$ directly to feature $X$ through an element-wise multiplication:

$$A = A \otimes X.$$  

It can be considered that $A$ serves as a class-specific and semantic meaningful attention map during the whole processing, to combine basic geometry information in the original features with augmented high-level semantic information.

Figure 5 shows the detailed architecture of our semantic activating module. Given the input features $X \in \mathbb{R}^{C \times H \times W}$, $F$ is achieved by a series of extra convolutional and pooling layers. To build this part of network, we refer to the standard SSD network structure from layer pool4 to fc7. Several modifications have been made to the network to expand its original local field of view and equip it with stronger representation ability. Concretely, we use three dilated convolutional layers [39] with $3 \times 3$ kernel size and dilated rate of 2, and one $3 \times 3$ dilated convolutional layer with dilated rate of 4. After that, another $1 \times 1$ convolutional layer is utilized to generate representation $F$. Two max pooling layers with kernel size of 3 and stride of 1 are deployed between them. The number of learned parameters in the above-mentioned five convolutional layers are kept unchanged so that they can get more sensible initialization from the pre-trained SSD model.

3.4 Multi-Task Joint Training

To take full advantage of the complementary strengths of the two proposed modules for object detection, we conduct a multi-task joint training to the whole network. The total loss function is defined as:

$$L = L_{\text{det}} + \lambda L_{\text{cam}},$$

where $L_{\text{det}}$ is the original object detection loss, $L_{\text{cam}}$ is the cross-entropy loss for the multi-label classification task in our semantic activating module:

$$L_{\text{cam}}(S, G) = -\sum_{i} S_{i} \cdot \log G_{i}.$$
Fig. 5 The detailed architecture of Semantic Activating Module, which contains a multi-label classification branch that takes deep representation $F$ as its input and encodes it with rich semantic knowledge. $F$ is then used to generate semantic meaningful convolutional response $A$ to activate the original input features.

$$- \sum_i (1 - S_i) \cdot \log(1 - G_i),$$

and $\lambda$ (empirically set to 0.1) is the loss weight to balance the two tasks.

4. Experiments

4.1 Implementation Details

We implement our ASAN based on the Caffe framework [40]. We use VGG16 based SSD [6] as our detection backbone. We build networks with two kinds of input dimensions of $300 \times 300$ and $512 \times 512$, namely ASAN300 and ASAN512, which are based on SSD300 and SSD512 respectively. The self-attending module contains 6 self-attending blocks for ASAN300 and 7 for ASAN512. We ignore the position attention branches in the last self-attending blocks in both networks since their feature maps already have the size of $1 \times 1$. The semantic activating module is attached to the first detection source layer (conv4_3) for both the low ($300 \times 300$) and high ($512 \times 512$) resolution networks.

The major parameters, including the reduction ratio $r$ in channel attention branch, maximum kernel size $n$ in position attention branch and the loss weight $\lambda$, are set to 8, 9, 0.1 in all our experiments. And we will discuss their influence later in Sect. 4.4.

4.2 PASCAL VOC

We conduct extensive experiments on the published object detection dataset PASCAL VOC which involves 20 categories. We report results on VOC 2007 test with models trained on the union of VOC 2007 trainval and VOC 2012 trainval. We also provide results on VOC 2012 test with models trained on the union of VOC 2007 trainvaltest and VOC 2012 trainval.

For fare comparison, we train all the models following the procedure of the original SSD experiments. We use a learning rate of $10^{-3}$ for 80k iterations, $10^{-4}$ for 20k iterations and $10^{-5}$ for another 20k iterations. We use SGD with a momentum of 0.9 and a weight decay of 0.0005. We use pre-trained SSD models on corresponding training set to initialize all the layers before conv6, and initialize the five extra convolutional layers in the semantic activating module using pre-trained parameters of conv5 to fc7 for better knowledge transfer. All the rest layers are randomly initialized with the xavier method [41].

Overall Performance. We compare our ASAN with several state-of-the-art detectors and summarize the results in Table 1. According to the results on VOC 2007 test, our ASAN achieves 80.0% and 81.7% mAP for low and high input dimensions respectively. We can see that our ASAN surpasses the original SSD [6] by a clear margin of 2.5% and 2.2% mAP gain for the two resolution networks. Also, using just VGG16 as our backbone, our ASAN significantly outperforms SSD [6] and DSSD [10], which are based on the more powerful ResNet101. Compared with recent work STDN [16], which introduces a scale-transfer module into the reverse connection network to do image super resolution and integrates it with DenseNet169, our ASAN achieves better performance for both input dimensions. Another advanced detector DES [36] pursues the similar goal of enriching semantics at low detection layers. In DES, a segmentation branch is trained to learn object information according to pixel-wise supervision. In ASAN, we exploit the remarkable localization ability of CNNs by training a generic classification branch. Using only image-level labels, ASAN shows apparent simplicity and is able to achieve comparable or even better performance than DES on both detection accuracy and speed.

We also compare our ASAN with representative two-stage methods. Our low resolution network ASAN300 already shows superior performance to most of the listed methods, including the classical two-stage detectors Fast R-CNN [22], Faster R-CNN [1] and the advanced version of Faster R-CNN [7] using ResNet101 as backbone, as well as
Table 1 Comparison of detection methods on PASCAL VOC dataset. The first part contains results of several representative two-stage detectors. The second and the third part presents results of popular one-stage methods with low and high input dimensions respectively.

| method               | backbone | framework | input size   | FPS    | mAP  |
|----------------------|----------|-----------|--------------|--------|------|
|                      |          |           |              |        | VOC 2007 | VOC 2012 |
| Fast R-CNN [22]      | VGG16    | Fast R-CNN | 1000 × 600   | 0.5    | 70.0 | 68.4 |
| Faster R-CNN [1]     | VGG16    | Faster R-CNN | 1000 × 600   | 7      | 73.2 | 70.2 |
| Faster R-CNN* [7]    | ResNet101| Faster R-CNN | 1000 × 600   | 2.4    | 76.4 | 73.8 |
| ION [12]             | VGG16    | Fast R-CNN | 1000 × 600   | 0.83   | 77.6 | 74.7 |
| MR-CNN [27]          | VGG16    | SSD       | 1000 × 600   | 0.03   | 78.2 | 73.9 |
| R-FCN [2]            | ResNet101| R-FCN     | 1000 × 600   | 9      | 80.5 | 77.6 |
| SSD300* [6]          | VGG16    | SSD       | 300 × 300    | 46     | 77.5 | 75.8 |
| SSD321 [6]           | ResNet101| SSD      | 321 × 321    | 11.2   | 77.1 | 75.4 |
| RON384++ [15]        | VGG16    | SSD       | 384 × 384    | 15     | 77.6 | 75.4 |
| DSSD321 [10]         | ResNet101| SSD      | 321 × 321    | 9.6    | 78.6 | 76.3 |
| STDN321 [16]         | DenseNet169 | SSD  | 321 × 321    | 28.4   | 79.3 | -   |
| DES300 [36]          | VGG16    | SSD       | 300 × 300    | 33.3   | 79.7 | 77.1 |
| ASAN300 (Ours)       | VGG16    | SSD       | 300 × 300    | 32.6   | 80.0 | 77.6 |
| SSD512* [6]          | VGG16    | SSD       | 512 × 512    | 19     | 79.5 | 78.3 |
| SSD513 [6]           | ResNet101| SSD      | 513 × 513    | 6.8    | 80.6 | 79.4 |
| DSSD513 [10]         | ResNet101| SSD      | 513 × 513    | 5.5    | 81.5 | 80.0 |
| STDN513 [16]         | DenseNet169 | SSD  | 513 × 513    | 20.3   | 80.9 | -   |
| DES512 [36]          | VGG16    | SSD       | 512 × 512    | 13.8   | 81.7 | 80.3 |
| ASAN512 (Ours)       | VGG16    | SSD       | 512 × 512    | 13.6   | 81.7 | 80.5†† |

†† http://host.robots.ox.ac.uk:8080/anonymous/J4ENIU.html

Our high resolution network ASAN512 produces 81.7% mAP, achieving state-of-the-art detection performance.

Same conclusions can be drawn according to the results on VOC 2012 test, which further suggests the great effectiveness of ASAN.

Inference Speed. We present the inference speed of our ASAN and all the competitors in the fifth column of Table 1. For fare comparison, all the runtime data is evaluated with batch size of 1 and computed on a Geforce GTX Titan X gpu (Maxwell architecture).

Results show that our ASAN300 runs at 32.6 FPS, which is much faster than all the two-stage detectors and is able to fulfill real-time detection. It is slower than original SSD [6] since the newly proposed modules bring about additional computation. However, compared with ResNet101 based SSD [6] and DSSD [10], our method runs much faster and meanwhile obtains higher mAP scores.

For the low resolution network, our ASAN achieves comparable detection speed with the state-of-the-art method DES [36] with higher accuracy and outperforms other advanced methods RON384++ [15] and STDN [16] in both detection speed and accuracy. Our high resolution network tends to require more inference time while the outstanding performance gain should be highlighted.

4.3 Evaluating Visualizations

Visualization of Self-Attending Module. In order to evaluate our main premise of introducing the self-attention mechanism, we first visualize some resulting attention maps generated by our self-attending module. In Fig. 6, we present our visualization results of four test examples. Intuitively, the channel attention branch should assign high weights to discriminative channels that detect specific visual patterns and depress those that contain noise or redundancy. From the first six rows in Fig. 6, we can see that identical expected results are obtained. The top weighted channels are able to identify discriminative semantic parts of different categories, while the lowest weighted channels either react to invalid background regions or contain few useful information for detection. To evaluate the position attention branch, we simply visualize the attention map from each block in the last row of Fig. 6. It can be seen that the crucial areas where objects exit are highlighted for special attention, while background regions are relatively weakened.

Qualitatively, the two attention branches in our self-attending module work as expected to learn what to emphasize or depress in a completely self-guided manner. They together encourage a more compact and semantic intensive distribution of the original features.

Visualization of Semantic Activating Module. For qualitative analysis how our semantic activating module performs on semantic exploiting, we also give some illustrations in Fig. 7. We can see that the original feature maps are relatively messier, which detect low-level geometric details, e.g. edges, corners and gradients in the image. On the contrary, the activation maps generated by our semantic activating module are more semantic meaningful, which demonstrate strong responses to discriminative parts of specific classes. It can be observed that after being activated by such a convolutional response, the activated feature maps tend to capture high-level semantic knowledge of objects with the visual details reserved. The results tell that our semantic activating module have successfully obtained desired effect of semantic augmentation.

other popular methods ION [12] and MR-CNN [27]. Our high resolution network ASAN512 produces 81.7% mAP, achieving state-of-the-art detection performance.

Same conclusions can be drawn according to the results on VOC 2012 test, which further suggests the great effectiveness of ASAN.

Inference Speed. We present the inference speed of our ASAN and all the competitors in the fifth column of Table 1. For fare comparison, all the runtime data is evaluated with batch size of 1 and computed on a Geforce GTX Titan X gpu (Maxwell architecture).

Results show that our ASAN300 runs at 32.6 FPS, which is much faster than all the two-stage detectors and is able to fulfill real-time detection. It is slower than original SSD [6] since the newly proposed modules bring about additional computation. However, compared with ResNet101 based SSD [6] and DSSD [10], our method runs much faster and meanwhile obtains higher mAP scores.

For the low resolution network, our ASAN achieves comparable detection speed with the state-of-the-art method DES [36] with higher accuracy and outperforms other advanced methods RON384++ [15] and STDN [16] in both detection speed and accuracy. Our high resolution network tends to require more inference time while the outstanding performance gain should be highlighted.

4.3 Evaluating Visualizations

Visualization of Self-Attending Module. In order to evaluate our main premise of introducing the self-attention mechanism, we first visualize some resulting attention maps generated by our self-attending module. In Fig. 6, we present our visualization results of four test examples. Intuitively, the channel attention branch should assign high weights to discriminative channels that detect specific visual patterns and depress those that contain noise or redundancy. From the first six rows in Fig. 6, we can see that identical expected results are obtained. The top weighted channels are able to identify discriminative semantic parts of different categories, while the lowest weighted channels either react to invalid background regions or contain few useful information for detection. To evaluate the position attention branch, we simply visualize the attention map from each block in the last row of Fig. 6. It can be seen that the crucial areas where objects exit are highlighted for special attention, while background regions are relatively weakened.

Qualitatively, the two attention branches in our self-attending module work as expected to learn what to emphasize or depress in a completely self-guided manner. They together encourage a more compact and semantic intensive distribution of the original features.

Visualization of Semantic Activating Module. For qualitative analysis how our semantic activating module performs on semantic exploiting, we also give some illustrations in Fig. 7. We can see that the original feature maps are relatively messier, which detect low-level geometric details, e.g. edges, corners and gradients in the image. On the contrary, the activation maps generated by our semantic activating module are more semantic meaningful, which demonstrate strong responses to discriminative parts of specific classes. It can be observed that after being activated by such a convolutional response, the activated feature maps tend to capture high-level semantic knowledge of objects with the visual details reserved. The results tell that our semantic activating module have successfully obtained desired effect of semantic augmentation.

other popular methods ION [12] and MR-CNN [27]. Our high resolution network ASAN512 produces 81.7% mAP, achieving state-of-the-art detection performance.

Same conclusions can be drawn according to the results on VOC 2012 test, which further suggests the great effectiveness of ASAN.

Inference Speed. We present the inference speed of our ASAN and all the competitors in the fifth column of Table 1. For fare comparison, all the runtime data is evaluated with batch size of 1 and computed on a Geforce GTX Titan X gpu (Maxwell architecture).

Results show that our ASAN300 runs at 32.6 FPS, which is much faster than all the two-stage detectors and is able to fulfill real-time detection. It is slower than original SSD [6] since the newly proposed modules bring about additional computation. However, compared with ResNet101 based SSD [6] and DSSD [10], our method runs much faster and meanwhile obtains higher mAP scores.

For the low resolution network, our ASAN achieves comparable detection speed with the state-of-the-art method DES [36] with higher accuracy and outperforms other advanced methods RON384++ [15] and STDN [16] in both detection speed and accuracy. Our high resolution network tends to require more inference time while the outstanding performance gain should be highlighted.
4.4 Ablation Study

To explore the contribution of each component in our ASAN to the final performance and discuss the impact of several major hyper-parameters, we conduct ablation study on PASCAL VOC dataset. All results in this section are tested on VOC 2007 test based on ASAN300 trained with VOC 2007 trainval and VOC 2012 trainval.

Effectiveness of Individual Components. In Table 2, we report results of several variants of ASAN to verify the effectiveness of each individual component. As indicated in Sect. 3.2, we propose self-attending module to produce semantic intensive detection source features by modeling global dependencies in both channel and spatial dimensions. To have a deep analysis how the self-attending mechanism enhance the original detector, we disassemble the module and present detection results of different network settings.
We can see that a single channel or position attention branch can obtain similar results, improving the mAP of the baseline (the original SSD) by 0.8% and 0.7%. With a combination of the two branches, the complete self-attending module achieves a relative improvement of 1.1%.

Further on, with the semantic activating module added, we can obtain another 1.4% mAP gain and reach the final result of 80.0%. The prominent advance proves the importance of low-level feature augmentation. It also provides evidence that our analogical use of the class activating mapping mechanism does help exploit high-level semantic knowledge for the object detection task.

**Impact of Loss Weight.** In Table 3, we conduct experiments to explore the impact of loss weight ($\lambda$ in Eq. (14)) between the detection task and the multi-label classification task. We train our complete ASAN300 network with different $\lambda$’s, i.e. 0.1, 0.5 and 1. Experimental results show that $\lambda = 0.1$ yields the lowest detection error and $\lambda = 0.5$ achieves relatively inferior performance. With $\lambda$ set to 1, which means to take the two tasks equally important, we get a high detection error up to 21.09%. Since we take object detection as our major task for evaluation, it is a good balance to set loss weight to be a small value of 0.1, so that the classification branch can act effectively on the main detection network without deviating the model too much.

**Reduction ratio and kernel size.** We also conduct experiments to determine the two hyper-parameters in the self-attending module, including the reduction ratio $r$ and the kernel size $n$. The reduction ratio $r$ determines the number of channels for the hidden layer in channel attention branch. In Table 3, we compare performance with four different values. The lowest detection error is achieved with the reduction ratio to be 8, which at the same time helps us keep the additional computation in budget. Larger values of 16 and 32 may lead to a loss of valid information. A reduction ratio of 4 seems to preserve more information, but may result in over-fitting of the model.

The kernel size $n$ determines the size of receptive field in the position attention branch. We conduct experiments by applying four different values: 5, 7, 9, and 11. Note that for each value, we refer to the largest kernel size applied to the lower detection layers (conv4_3 and fc7), kernel sizes for higher layers are adjusted correspondingly. According to the results in Table 3, we can see that the performance gets improved with larger kernel sizes and the best performance can be achieved with the maximum kernel size set to be 9, which proves the benefit of dense connections by large filters. However, we obtain a relatively high detection error of 20.16% with the largest kernel size of 11. We conjecture this result also as model over-fitting since the training losses converge in all the cases of different kernel sizes.

### 4.5 Microsoft COCO

To further validate the proposed ASAN and to better evaluate its performance on multi-scale detection, we carry out experiments on the MS COCO dataset. We use trainval35k set for training and report results on test-dev2017 set. All the models are trained with an initial learning rate of $10^{-3}$ for 280k iterations, $10^{-4}$ for 80k iterations and $10^{-5}$ for another 40k iterations. We keep other settings consistent with those in PASCAL VOC experiments.

In Table 4, we compare our ASAN with several representative baselines. It can be seen that ASAN produces

### Table 3 Ablation study of several major hyper-parameters, including loss weight $\lambda$, reduction ratio $r$ and kernel size $n$.

| method | independent variables | value | params | error |
|--------|-----------------------|-------|--------|-------|
| ASAN300 | loss weight ($\lambda$) | 0.1  | 35.53M | 19.97 |
|         |                       | 0.5  | 35.53M | 20.87 |
|         |                       | 1    | 35.53M | 21.09 |
|         | reduction ratio ($r$) | 4    | 35.97M | 20.35 |
|         |                       | 8    | 35.53M | 19.97 |
|         |                       | 16   | 35.31M | 20.11 |
|         |                       | 32   | 35.20M | 20.45 |
|         | kernel size ($n$)     | 5    | 34.43M | 20.49 |
|         |                       | 7    | 35.47M | 20.21 |
|         |                       | 9    | 35.53M | 19.97 |
|         |                       | 11   | 35.59M | 20.16 |

### Table 4 Comparison of detection methods on MS COCO dataset.

| method          | backbone | data       | Avg. Precision, IoU: | Avg. Precision, Area: |
|-----------------|----------|------------|-----------------------|------------------------|
|                 |          |            | 0.5/0.95 | 0.5 | 0.75 | S | M | L   |
| Faster R-CNN [1] | VGG16    | trainval  | 24.2  | 45.3 | 23.5 | 7.7 | 26.4 | 37.1 |
| Faster R-CNN* [7]| ResNet101 | trainval  | 34.9  | 55.7 | 37.4 | 15.6 | 38.7 | 50.9 |
| R-FCN [2]       | ResNet101 | trainval  | 29.9  | 51.9 | -    | 10.8 | 32.8 | 45.0 |
| SSD300* [6]     | VGG16    | trainval35k| 25.1  | 43.1 | 25.8 | 6.6 | 25.9 | 41.4 |
| SSD321 [6]      | ResNet101| trainval35k| 28.0  | 45.4 | 29.3 | 6.2 | 28.3 | 49.3 |
| ASAN300 (Ours)  | VGG16    | trainval35k| 28.2  | 47.2 | 29.3 | 8.4 | 29.6 | 45.3 |
| SSD512 [6]      | ResNet101| trainval35k| 28.8  | 48.5 | 30.3 | 10.9 | 31.8 | 45.3 |
| ASAN512 (Ours)  | VGG16    | trainval35k| 31.2  | 50.4 | 33.3 | 10.2 | 34.5 | 49.8 |
|                 |          |            | 32.3  | 52.4 | 34.2 | 13.0 | 35.8 | 47.6 |
28.2% AP and 32.3% AP for low and high input dimensions, which surpasses the baseline score of SSD with a large margin of 3.1% and 3.5%. It also outperforms ResNet101 based SSD, which is deeper and much slower.

Moreover, ASAN performs especially well on small objects due to our semantic activating strategy to lower layers. ASAN improves the baseline score of SSD300 for small objects from 6.6% to 8.4% and that of SSD512 from 10.9% to 13.0%, yielding relative improvements up to 27.3% and 19.2%. For larger objects, our ASAN also outperforms the baseline SSD method by a clear margin, but it is slightly worse than ResNet101 based SSD since deeper networks have the advantage on wider range detection.

5. Conclusion

In this paper, we propose a novel self-attending and semantic activating network (ASAN) for single-shot object detection, which consists of a self-attending module for semantic enhancement by learning channel and position wise global feature dependencies in a self-guided manner, and a semantic activating module for explicit semantic activation by exploiting high-level semantic knowledge leveraging the class activation mapping mechanism. Qualitative evaluation verifies the effectiveness of the two modules. We conduct extensive experiments on two popular benchmarks, PASCAL VOC and MS COCO. The experimental results show that ASAN achieves an outstanding performance compared with state-of-the-art methods, fulfilling a high accuracy real-time detection.

References

[1] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” Advances in neural information processing systems, vol.39, no.6, pp.91–99, 2015.
[2] J. Dai, Y. Li, K. He, and J. Sun, “R-fcn: Object detection via region-based fully convolutional networks,” Advances in Neural Information Processing Systems, pp.379–387, 2016.
[3] M. Everingham, L. Van Gool, C.K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” International journal of computer vision, vol.88, no.2, pp.303–338, 2010.
[4] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C.L. Zitnick, “Microsoft coco: Common objects in context,” European conference on computer vision, vol.8693, pp.740–755, Springer, 2014.
[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” Proceedings of the IEEE conference on computer vision and pattern recognition, pp.779–788, 2016.
[6] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A.C. Berg, “Ssd: Single shot multibox detector,” European conference on computer vision, vol.9905, pp.21–37, Springer, 2016.
[7] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” Proceedings of the IEEE conference on computer vision and pattern recognition, pp.777–787, 2016.
[8] C. Szegedy, S. Ioffe, V. Vanhoucke, and A.A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” Thirty-First AAAI Conference on Artificial Intelligence, 2017.
[9] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, “Speed/accuracy trade-offs for modern convolutional object detectors,” Proceedings of the IEEE conference on computer vision and pattern recognition, pp.7310–7311, 2017.
[10] C.Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A.C. Berg, “Dssd: Deconvolutional single shot detector,” arXiv preprint arXiv:1701.06659, 2017.
[11] Z. Cai, Q. Fan, R.S. Fertis, and N. Vasconcelos, “A unified multi-scale deep convolutional neural network for fast object detection,” European conference on computer vision, vol.9908, pp.354–370, Springer, 2016.
[12] S. Bell, C.L. Zitnick, K. Bala, and R. Girshick, “Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks,” Proceedings of the IEEE conference on computer vision and pattern recognition, pp.2874–2883, 2016.
[13] T. Kong, A. Yao, Y. Chen, and F. Sun, “Hypernet: Towards accurate region proposal generation and joint object detection,” Proceedings of the IEEE conference on computer vision and pattern recognition, pp.845–853, 2016.
[14] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.2117–2125, 2017.
[15] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen, “Ron: Reverse connection with objectness prior networks for object detection,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.5936–5944, 2017.
[16] P. Zhou, B. Ni, C. Geng, J. Hu, and Y. Xu, “Scale-transferable object detection,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.528–537, 2018.
[17] S. Zhang, L. Wen, X. Bian, Z. Lei, and S.Z. Li, “Single-shot refinement neural network for object detection,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.4203–4212, 2018.
[18] S. Liu, D. Huang, and Y. Wang, “Receptive field block net for accurate and fast object detection,” Proceedings of the European Conference on Computer Vision (ECCV), vol.11215, pp.404–419, 2018.
[19] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Object detectors emerge in deep scene cnns,” arXiv preprint arXiv:1412.6856, 2014.
[20] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” Proceedings of the IEEE conference on computer vision and pattern recognition, pp.2921–2929, 2016.
[21] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” Proceedings of the IEEE conference on computer vision and pattern recognition, pp.580–587, 2014.
[22] R. Girshick, “Fast r-cnn,” Proceedings of the IEEE international conference on computer vision, pp.1440–1448, 2015.
[23] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” Proceedings of the IEEE international conference on computer vision, pp.2901–2909, 2017.
[24] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” arXiv preprint arXiv:1312.6229, 2013.
[25] J. Redmon and A. Farhadi, “Yolo9000: better, faster, stronger,” Proceedings of the IEEE conference on computer vision and pattern recognition, pp.7263–7271, 2017.
[26] A. Shrivastava and A. Gupta, “Contextual priming and feedback for faster r-cnn,” European Conference on Computer Vision, vol.9905, pp.330–348, Springer, 2016.
[27] S. Gidaris and N. Komodakis, “Object detection via a multi-region activating network (ASAN) for single-shot object detection,” Proceedings of the IEEE International Conference on Computer Vision, pp.1134–1142, 2015.
[28] Y. Zhu, C. Zhao, J. Wang, X. Zhao, Y. Wu, and H. Lu, “Couplenet: Coupling global structure with local parts for object detection,” Proceedings of the IEEE International Conference on Computer Vision, pp.4126–4134, 2017.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” Advances in Neural Information Processing Systems, pp.5998–6008, 2017.

[30] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, “Residual attention network for image classification,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.3156–3164, 2017.

[31] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” Proceedings of the IEEE conference on computer vision and pattern recognition, pp.7132–7141, 2018.

[32] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei, “Relation networks for object detection,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.3588–3597, 2018.

[33] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, “Cham: Convolutional block attention module,” Proceedings of the European Conference on Computer Vision (ECCV), vol.11211, pp.3–19, 2018.

[34] J. Park, S. Woo, J.Y. Lee, and I.S. Kweon, “Barn: Bottleneck attention module,” arXiv preprint arXiv:1807.06514, 2018.

[35] J. Fu, J. Liu, H. Tian, Z. Fang, and H. Lu, “Dual attention network for scene segmentation,” arXiv preprint arXiv:1809.02983, 2018.

[36] Z. Zhang, S. Qiao, C. Xie, W. Shen, B. Wang, and A.L. Yuille, “Single-shot object detection with enriched semantics,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.5813–5821, 2018.

[37] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv preprint arXiv:1409.1556, 2014.

[38] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun, “Large kernel matters – improve semantic segmentation by global convolutional network,” Proceedings of the IEEE conference on computer vision and pattern recognition, pp.4353–4361, 2017.

[39] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," arXiv preprint arXiv:1511.07122, 2015.

[40] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” Proceedings of the 22nd ACM international conference on Multimedia, pp.675–678, ACM, 2014.

[41] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” Proceedings of the 13th Conference on Artificial Intelligence and Statistics, pp.249–256, 2010.

Xinyu Zhu received her B.S. degree in microelectronic science and engineering from Fudan University of China in 2016. She is currently a graduate student in the State-key Lab of ASIC and System at Fudan University, China. Her research interests include object detection, fine-grained image classification and embedded software.

Jun Zhang received his B.S. degree in microelectronic science and engineering from Fudan University of China in 2016. He is currently a graduate student in the State-key Lab of ASIC and System at Fudan University, China. His research interests include computer vision, machine learning and embedded software.

Gengsheng Chen received his Master and PhD degrees in electronic engineering from Fudan University of China. He is now a Senior Research Engineer in the State-key Lab of ASIC and System, Fudan University, China. He served Motorola Electronics, Nortel Networks, Siemens Technical Innovation Center and Zarlink Semiconductor between 1994 and 2005. His current major research interests include image processing, embedded system, FPGA circuit and system. He has over 30 papers published on international academic conference and journals, he holds five Chinese patents.