applied sciences

MDPI

Article

Lightweight Attention Pyramid Network for Object Detection and Instance Segmentation

Jiwei Zhang [1], Yanyu Yan [2], Zelei Cheng [3] and Wendong Wang [1,2,*]

1 School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China; zhangjiwei5510@163.com
2 State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China; ibuptyanyy@foxmail.com
3 The Center for Education and Research in Information Assurance and Security (CERIAS), Purdue University, West Lafayette, IN 47906, USA; cheng473@purdue.edu
* Correspondence: wdwang@bupt.edu.cn

Received: 7 December 2019; Accepted: 13 January 2020; Published: 28 January 2020

Abstract: Feature pyramids of convolutional neural networks (ConvNets)—from bottom to top—are used by most recent researchers for the improvement of object detection accuracy, but they seldom aim to address the correlation of each feature channel and the fusion of low-level features and high-level features. In this paper, an Attention Pyramid Network (APN) is proposed, which mainly contains the adaptive transformation module and feature attention block. The adaptive transformation module utilizes the multiscale feature fusion, and makes full use of the accurate target location information of low-level features and the semantic information of high-level features. Then, the feature attention block strengthens the features of important channels and weakens the features of unimportant channels through learning. By implementing the APN in a basic Mask R-CNN system, our method achieves state-of-the-art results on the MS COCO dataset and 2018 WAD database without bells and whistles. In addition, the structure of the APN makes the network parameters lighter, and runs at 4 ms on average, which is ignorable when compared to the inference time of the backbone of ConvNet.

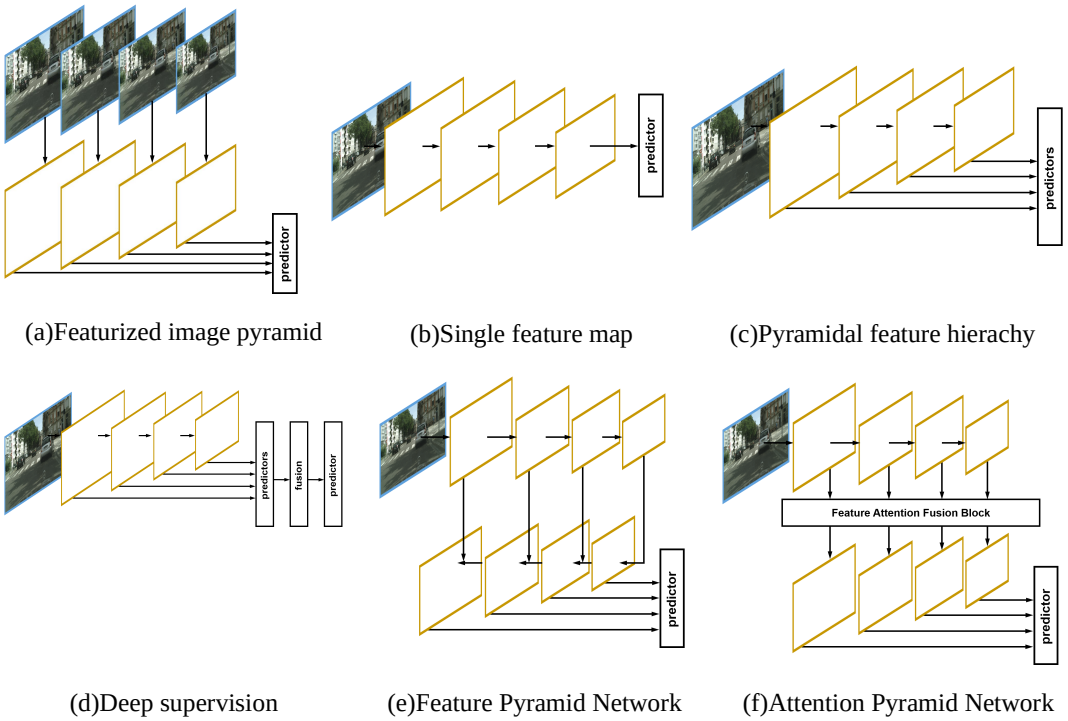
Keywords: APN; adaptive transformation; channel-wise attention; object detection; instance segmentation

1. Introduction

Along with the popularization of the artificial intelligence systems [1,2], IOT [3] and the accumulation of image data [4], automatic object detection is increasingly being widely used in video surveillance and robot vision. Compared with manual inspections, computer vision technology can effectively improve the inspection efficiency and avoid the influence of subjectivity on accuracy. Object detection is a fundamental computer-vision task [5], and the existence of multiple scales and ratios is the most challenging problem in object detection. More and more attention is being paid to this problem, and various detection methods have emerged. In the image pyramid methods [6,7], as shown in Figure 1a, images are generally resized to multiple scales and then resized to the same ratio for training and inference. Because of the large number of images, the methods are computationally expensive.

By using pyramids of reference boxes (which are called anchors), the multiple scales and ratios within the same feature map size are addressed and do not cause extra computational complexity for the multiple feature map sizes [8]. As shown in Figure 1b, a trade-off between the anchors is made. The anchor described above is adopted by methods such as Faster R-CNN [8], strengthened RPN [9], and region-based fully convolutional networks [10]. The anchors with different scales and ratios are

created in order to predict candidate bounding boxes for multi-scale objects. However, the topmost feature map has a limitation on its fixed receptive field size—this method struggles to detect too large or too small objects.



(a)Featurized image pyramid (b)Single feature map (c)Pyramidal feature hierachy

(d)Deep supervision (e)Feature Pyramid Network (f)Attention Pyramid Network

Figure 1. Different architectures for addressing multiple scales and sizes where feature maps are indicated by yellow outlines.

Recent object detection methods, such as single shot multibox detector (SSD) [11] and multi-scale deep convolutional neural network (MS-CNN) [12] are used for solving the object detection of multiple scales and ratios. As shown in Figure 1c, the feature pyramid from bottom to top is adopted; a well-directed classifier and bounding box regressor [13] with multiple scales/sizes are trained to handle the object detection of different scales and ratios. However, owing to the multiple types of bounding box regressors, we should fit each with specific training data. This method would necessitate dividing the original training data into multiple types, and each filter would then be fed much less training data, so the division cannot guarantee adequate training data for each filter. Besides, each filter requires different proposal sizes, resulting in implementational complexity and the burden of network training.

As shown in Figure 1d, using sideways predictions with deep supervision is another decent method to use feature maps from different stages. Such a method generates the final prediction by fusing all of the sideways predictions [14,15]. A top-down structure, such as that in Figure 1e, has recently become popular. This structure has been proven to work well in feature pyramid networks (FPN) [16], deconvolutional single shot detector (DSSD) [17], and Mask R-CNN [18]. Fusing feature maps layer by layer is not efficient or effective enough when there are many layers to be combined together. FPN developed a top-down pathway with lateral connections to build high-level semantic feature maps at all scales. This architecture shows significant improvements as a generic feature extractor in several applications such as object detection and instance object segmentation. However, the FPN module only takes neighboring feature maps into consideration. Adding the feature maps from top to bottom, level by level, weakens the influence from the top feature map to the bottom feature map.

However, these methods seldom both use the relationship of each feature channel and the fusion of low-level features and high-level features [19]. In this paper, we propose an upgraded method,

the Attention Pyramid Network (APN), which can utilize both multiscale feature fusion and the channel-wise attention mechanism to generate new feature pyramids. Instead of only fusing the neighboring feature maps, we designed a new sufficient module that takes all levels of the feature maps into consideration by using adaptive transformation, which shortens the information propagation path of multiple feature maps to generate a new feature pyramid. Then, our network focuses on feature attention [20], which reweights different channels of the feature maps. Different from the traditional adaptive transformation and feature attention block, the proposed APN can utilize both multiscale feature fusion and channel-wise attention mechanisms to generate new feature pyramids. Moreover, the proposed APN is a lightweight network. The modification is independent of the backbone convolutional architecture, and, in this paper, we present our result based on ResNets [21]. We evaluated our method on the 2018 CVPR WAD dataset and MS COCO dataset [22]. Without any bells and whistles, we achieve a better result than the baseline methods. Moreover, our module achieves the above improvement in a very short detection time, and our code and models will be made publicly available. The contributions of this paper are summarized as follows:

- The adaptive transformation module can increase the information interaction between multiscale features and improve the performance of feature pyramids.
- The feature attention block adopt a channel-wise attention mechanism to generate new feature pyramids, and make full use of the relationship between channels. In addition, it can strengthen the important channel information and weaken the unimportant channel information.
- A novel APN is proposed, and it is lightweight. The performance of the proposed APN was evaluated via extensive simulations using the MS COCO and the WAD datasets; the results show the effectiveness of our approach.

2. Related Work

Feature fusion is often used for image classification, visual tracking, object detection, etc. A single classifier usually cannot handle some complex data classification; the technology of feature fusion is used to fuse different visual features to solve the problem of image classification [23]. In the problem of visual tracking, lighting, occlusion, and other factors often affect the recognition effect. Some association models based on sparse representation [24,25] have been proposed for feature fusion and have a certain robustness in visual tracking. In recent years, some algorithms based on deep learning have adopted the idea of feature fusion. Song et al. proposed a deep feature fusion network for the classification of hyperspectral images [26]. Using residual learning to optimize the convolutional layer, and the model to fuse the features of different layers, the method achieved better results than other classifiers. To solve the complex situation such as the small size of the object in the aerial image, a feature fusion deep network was proposed [27], and the spatial relationship between objects was increased by the fusion of network layer features, thus accurate detection results were obtained. Facial expressions are very important information in human behavior research. To recognize three-dimensional faces, Tian et al. proposed a deep feature fusion convolutional neural networks (CNN) [28], which combines different two-dimensional face information to fine-tune the network model, and achieves effective detection results. In addition, Starzacher et al. combined artificial neural networks and support vector machines to achieve feature fusion and applied it to embedded devices [29]. It can be used for vehicle classification and tracking in traffic monitoring, and has achieved good execution time and classification rate on embedded platforms.

Attention was first introduced in the NLP area. Generally, attention is intended to choose significant feature representations at some specific locations. Luong et al. [30] proposed global and local attention approaches for neural machine translation. The global approach pays attention to all source words while the local one only focuses on the subset of source words at a time. Vaswani et al. [31] proposed a self-attention method for machine translation; the encoder and decoder can be connected via an attention mechanism. However, it then gains great popularity in the computer vision area. Wang et al. [32] proposed residual attention network where attention modules are stacked to generate

attention-aware features. Moreover, attention residual learning is proposed to train very deep networks. Fan et al. [33] developed a few-shot object detection model with an attention module on RPN and a detector. Thus, the irrelevant background boxes can be filtered out. Wang et al. [34] proposed a novel domain-attention mechanism for a universal object detection model. A domain-attention module is leveraged and enables adapters to specialize on some individual domains. Zhang et al. [35] proposed a backward attention filter to help the region proposal network generate reasonable regions of interest (ROIs). The informative features are emphasized and, by utilizing the semantic features from the deep layers, it can suppress the distractive features. Li et al. [36] sequentially integrated different kinds of soft attention into CNNs for single-stage object detection. Huang et al. [37] designed Inverted Attention to improve object detection. It inverts attention to feature maps and assigns more attention to complementary object parts.

3. Materials and Methods

3.1. The Framework

As bottom-up feature maps are characterized with lower-level semantics, their activations are accurately localized since they are subsampled fewer times. Contrarily, top-down feature maps are enriched with abundant semantic information because of the multiple levels of the processing [16]. Besides, we aim to address the correlation of each feature channel to improve the accuracy of object recognition. In this paper, we propose the APN based on the feature maps fusion paradigm and channel-wise attention mechanism. The feature fusion module takes advantage of different scales of feature maps to generate new pyramidal feature maps. This can help the following detectors to utilize the contextual information. In our feature attention block, we strengthen the features of important channels and weaken the features of unimportant channels. We focus on Mask R-CNN [18] to enhance a greater understanding of the general-purpose generated APN. In Figure 2, the architecture of our proposed APN is shown. The feature attention fusion block contains the adaptive transformation module, feature attention block, and the fusion block.

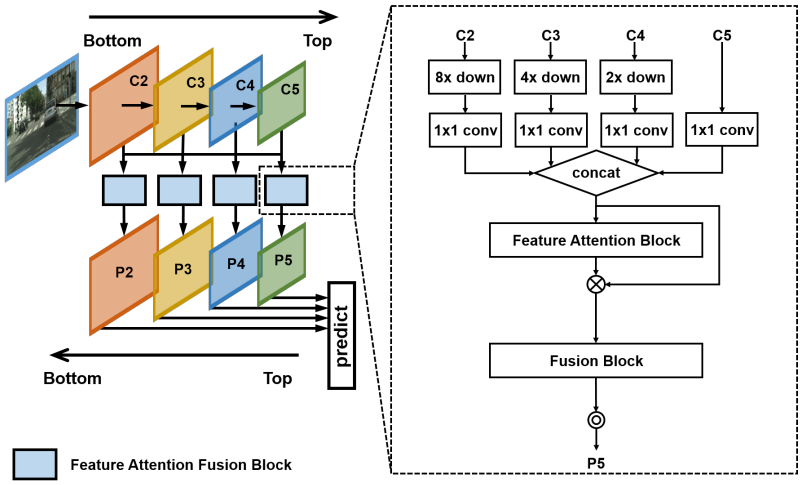


Figure 2. The architecture of our proposed Attention Pyramid Network (APN). A building block illustrates the feature attention fusion block, which takes the P_5 generation process as an example.

First, we select an arbitrary size single-scale image as an input. Then, we generate proportionally sized feature maps using backbone of the convolutional architectures. We use ResNets [21] as the backbone to compute a feature hierarchy that consists of feature maps at several scales with a scaling stride step of 2 pixels. In the feedforward computation of ConvNet, many layers producing output maps of the same size are available, in which these layers are in the same network stage. As shown in Figure 2, we use the feature activations output using each stage of the last residual block, denoted

as $\{C_2, C_3, C_4, C_5\}$ for the outputs of conv2, conv3, conv4, and conv5 with the corresponding strides of {4, 8, 16, 32} pixels with respect to the input images, respectively. However, we do not integrate conv1 into the pyramid because of its large memory consumption. As shown in Figure 2, we define the relative concept of the bottom feature maps and the top feature maps.

3.2. Adaptive Transformation Module

Improvement in object detection accuracy requires building high-level semantic feature maps at all scales for improved bottom-up and top-down pathways. To solve the problems associated with different resolutions of the feature maps in different network stages, we adopt an adaptive transformation to the feature maps separately. Specifically, we take C_5 as an example. As discussed in the previous section, all the other levels of the feature maps are considered as bottom feature maps to C_5. To resize all the bottom feature maps to the same C_5 size, all bottom feature maps are downsampled using nearest neighbor downsampling for the simplicity of different factors. We define the downsampling as resolution downsampling, in which the spatial resolution is C_4 by a stride factor of 2 pixels. Moreover, we downsample the spatial resolution of C_3 by a factor of 4, and so on. Specifically, the downsampling factor should be adjusted according to the feature maps strides corresponding to the input image. Contrarily, when we adjust the resolution of the top feature maps to the bottom feature maps, we upsample the top feature maps, using nearest neighbor downsampling. For example, when we take all the feature maps to generate the feature maps of the same size (C_4), we upsample the C_5 spatial resolution by the factor of 2.

Because all levels of the pyramid use shared classifiers/regressors as in FPN, we fix the numbers of channels (denoted as d) in all feature maps. We set $d = 256$, and all extra convolutional layers have 256-channel outputs. To fulfill this design, we apply a 1×1 convolutional layer to all the transformed feature maps to fix the numbers of the channels. Then, we concatenate all the feature maps into the same size. We also develop the method to fuse the selected feature maps—feature attention block—to generate pyramid features.

3.3. Feature Attention Block

The feature attention block reweights the concatenated feature maps using a channel-wise interaction. As described above, each channel of the concatenated feature map is obtained from the separate convolutional filter. Each convolutional filter will compute over the previous feature map to generate one channel of the concatenated feature map. It indicates that the convolutional filters will not interact with each other. The convolutional filters are irrelevant to each other, which means that the channels of the concatenated feature map are irrelative to each other. Therefore, the concatenated feature map is the lack of channel correlation. We design the channel-based attention fusion module to generate the final feature map. Figure 3 shows how we develop the channel-wise interaction on the concatenated feature maps.

We squeeze all spatial information into one pixel, in which each channel is treated as the representation of the whole channel's information. We adopt a global average pooling as a simple method to implement this idea. Therefore, the original shape of the concatenated feature map is squeezed to the shape of $N \times 1 \times 1 \times 1024$ (batch size $\times$ height $\times$ width $\times$ channel). Here, we fix the channel amount of the concatenated feature map to 1024, which is denoted as F_{sq}.

We parameterize a gating mechanism by forming a bottleneck [38] with two 1×1 convolutional layers to limit the module computation complexity. We apply the non-linearity after each 1×1 convolutional layer. We set the bottleneck ratio r to 16 to reduce the numbers of channels, and the first 1×1 convolutional layer is applied to the bottleneck ratio to decrease the dimensions of the channel-wise static that arises from F_{sq} after processing. To train the reweighted channel-wise statistics capable of learning nonlinear interaction between channels, we apply the statistics with a nonlinear activation ReLU. Then, we adopt a 1×1 convolutional layer with the same bottleneck ratio to increase the dimensions. Since we already concatenate all the feature maps to process such channel-wise

interaction, we allow multiple channels to be emphasized rather than a one-hot activation. To fulfill the learning of a non-mutually-exclusive relationship, we apply a nonlinear activation sigmoid. We make the output tensor and input as vector s the same shape. This process is denoted as F_{ex}.

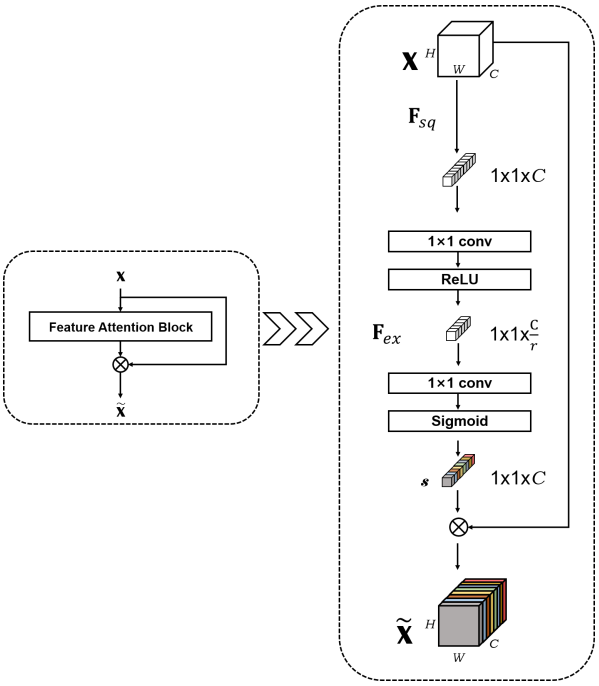


Figure 3. Illustration of our feature attention block. Note that we display our feature map without the dimension of batch size for easier understanding.

In this paper, we use the 1×1 convolutional layer to replace the fully-connected layers—instead of using the complex SENet [39] structure—to implement the channel-wise interaction. Our implemented structure has a lighter module that proved effective in the experiment presented in Section 4. After obtaining the vector s, we apply a rescaling to the concatenated feature maps, implemented by adopting channel-wise multiplication between the concatenated feature maps and the vector. In this way, we obtain the channel-wise attention feature maps.

3.4. Fusion Block

Here, we propose two different ways to fuse the reweighted feature map. One method is the convolutional fusion, and we choose to implement the fusion block by applying a 3×3 convolutions to each feature map. This convolution layer will fuse the 1024 channels feature map to the 256 channel feature map, and reduce the complexity of the parameters. The other method we came up with was the fusion block with a simple channel-wise addition fusion. This method works by dividing the feature maps into four separate maps according to our fixed channel number based on 256 different stages of the feature maps. After the processing, four reweighted feature maps from different stages of the backbone are obtained. We adapt the fusion operation to the four reweighted feature maps to generate one feature map, implementing this operation by channel-wise addition. According to the result of the experiment presented in Section 4, we chose the first method as our final model design.

Finally, we append a 3×3 convolutions on each merged map to generate the final feature maps and to reduce the aliasing effect of upsampling and downsampling. This final set of feature maps is called $\{P_2, P_3, P_4, P_5\}$, corresponding to $\{C_2, C_3, C_4, C_5\}$ of the same spatial sizes. Specifically, each feature map in our final set is generated from all source feature maps with a better channel-wise reweighted fusion.

3.5. Loss Function

During the training of the RPN [8], each anchor was assigned a binary class label and five parametric coordinates. To train the RPN, we needed to find positive and negative samples from all anchors. The positive samples anchors were needed to satisfy the following conditions: the Intersection over Union (IoU) [8] overlap between an anchor and the ground-truth is greater than 0.7, or an anchor has the highest IoU overlaps with a ground-truth. Negative samples were defined as: IoU overlap less than 0.3 to all ground truth boxes. Anchors that are neither positive nor negative were discarded. We use the multi-task loss to minimize the objective function, which is defined as follows:

$$L(p_k, b_k) = \frac{1}{N_{gtbox}} \sum_k (L_{clas}(p_k, p_k^*) + \frac{\lambda}{N_{anchor}} \sum_k p_k^* L_{reg}(b_k, b_k^*) \quad (1)$$

where k represents the index of an anchor in the mini-batch. N_{gtbox} denotes the number of ground truth boxes and N_{anchor} denotes the number of anchors. p_k is the predicted possibility that anchor k is the target object. p_k^* is a binary value asserting whether the anchor is positive. b_k is a vector that is used to represent the predicted bounding box while b_k^* denotes the ground-truth bounding box.

In the multi-task loss, we have two parts of the loss function. The first one is the classification loss. Here, we use a cross-entropy function, which is defined as:

$$L_{clas}(p_k, p_k^*) = p_k^* log(p_k) \qquad (Cross\ Entropy) \quad (2)$$

The second part is the regression loss. We need to force the distance between the predicted bounding box and the ground truth to be as close as possible. Here, we measure the distance by smooth L1-norm, which is

$$L_{reg}(b_k, b_k^*) = \begin{cases} |b_k - b_k^*|, if \quad |b_k - b_k^*| < 1 \\ (b_k - b_k^*)^2, \text{otherwize} \end{cases} \text{(smooth L1)} \quad (3)$$

The term $p_k^* L_{reg}(b_k, b_k^*)$ means only when the anchors are positive should the regression loss function be activated, i.e., $p_k^* = 1$. The outputs of the classification and regression layers include p_k and t_k, respectively. Then, they need to be normalized with N_{clas} and N_{reg}, and balanced by weight λ.

For regression, the parameterizations of the four coordinates (top left corner, width, and height) are defined as:

$$\begin{cases} b_x = \frac{x - x_a}{w_a} \\ b_y = \frac{y - y_a}{h_a} \\ b_w = log(\frac{w}{w_a}) \\ b_h = log(\frac{h}{h_a}) \end{cases} \quad \begin{cases} b_x^* = \frac{x^* - x_a}{w_a} \\ b_y^* = \frac{y^* - y_a}{h_a} \\ b_w^* = log(\frac{w^*}{w_a}) \\ b_h^* = log(\frac{h^*}{h_a}) \end{cases} \quad (4)$$

Here, x, y, w, h denote the four coordinates of the predict box; x_a, y_a, w_a, h_a denote those of the anchor box; and x^*, y^*, w^*, h^* denote those of the ground truth box.

3.6. Implementation Details

We re-implemented Mask R-CNN [18] and FPN [16] based on TensorFlow [40]. All pre-trained models used in the experiments are publicly available. We trained our model using image centric training [41]. For each training image, we sampled 200 Rois [8] with a positive-to-negative ratio of 1:3. All pre-trained models that we used in experiments are publicly available. We replaced the FPN with the APN during training, and the corresponding models were pre-trained from ImageNet.

We adopted the same end-to-end training as in the Mask R-CNN; however, we chose slightly different hyperparameters. Following Mask R-CNN, proposals were generated from an independently trained RPN [8,18] to allow convenient ablation and fair comparison. We took two images in one image batch for training and used two NVidia TitanX GPUs (one image per GPU).

4. Results and Discussion

We comprehensively evaluated the APN using the MS COCO dataset and the 2018 WAD dataset, and our results outperform the baseline, i.e., the original FPN. In addition, The parameters and running time of the APN are presented to prove how lightweight the network is, and we use the legend to show the actual effect of object detection and segmentation. To prove the validity of the adaptive transformation and feature attention block, we conducted comparative experiments on them separately. We present our results following the standard evaluation metrics [18], denoted as Intersection over Union (IoU) and Average Precision (AP), respectively, as well as using the instance segmentation average precision, denoted as AP^M.

4.1. Dataset and Metrics

The MS COCO dataset is one of the most challenging datasets for object detection task due to data complexity. It consists of 115 k images for training and 5 k images for validation (new split of 2017). In total, 20 k images were used in the test-dev and 20 k images were used as a test-challenge. Ground-truth labels of both test-challenge and test-dev are not publicly available. The MS COCO dataset has 80 classes with bounding box annotation. We trained our models on the train-2017 subset and report results on the val-2017 subset for ablation study. We also report results for the test-dev for comparison with the single model result.

In addition, the 2018 WAD dataset comes from the 2018 CVPR workshop on autonomous driving, which was sponsored by Baidu Inc. This dataset consists of approximately 35 K images for training and 5 K images for validation. In total, 2 K test images were used in the test challenge. The dataset includes seven classes with pixel-wise instance mask annotation. Ground-truth labels of the test challenge are not publicly available. We trained our models on the training dataset and report our results for the test-dev for comparison.

We followed the standard evaluation metrics, i.e., IoU,AP, $AP_{@0.5}$, $AP_{@0.75}$, AP_S, AP_M, and AP_L. The last three measure performance corresponding to objects with different scales. Since our framework is general to both object detection and instance segmentation, we also report the result for instance segmentation.

Below is the definition of IoU:

$$IoU = \frac{area(B_p \cap B_{GT})}{area(B_p \cup B_{GT})}, \tag{5}$$

where B_p is the prediction region from the detection/segmentation network and B_{GT} is the ground-truth body region. We used the mean IoU calculated from all test images.

The AP is defined as follows:

$$AP = \frac{1}{N} \sum^{all} \frac{tp}{(tp + fp)}, \tag{6}$$

where tp means the true positive, fp means the false positive, and N means the number of the detection/segmentation results. AP in Equation (6) represents the averaged value of all categories. Traditionally, this is called "mean average precision" (mAP). In the MS COCO dataset, it makes no distinction between AP and mAP. Specifically, the MS COCO dataset uses 10 IoU thresholds of 0.50:0.05:0.95. This is a break from tradition, where AP is computed at a single IoU of 0.5 (which corresponds to our metric $AP_{.50}$).

4.2. Experiment on the MS COCO Dataset

We experimented on the Mask R-CNN [18] baselines, re-implemented based on the Tensorflow. All pre-trained models that we used in experiments are publicly available. We replaced FPN with APN during training, where the corresponding models were pre-trained from ImageNet. We adopted the same end-to-end training as in the Mask R-CNN; however, we chose slightly different hyperparameters.

We took two images in one image batch for training and used two NVidia TitanX GPUs (one image per GPU). The shorter and longer edges of the images were 800 and 1333, if not otherwise noted. For object detection, we trained our model with a learning rate that starts from 0.01, which decreased by a factor of 0.1 after 960 k and 1280 k iterations and finally terminated at 1440 k iterations. This training schedule resulted in 12.17 epochs over the 118,287 images in the MS COCO 2017 training dataset. The rest of the hyperparameters remained the same as the Mask R-CNN.

We evaluated the performance of the proposed APN on the MS COCO dataset and compared the `test-dev` results with the recent state-of-art models with different feature map fusion methods. The results of our proposed ResNet-50 based APN are presented in Table 1. As shown in Table 1, our model, trained and tested on single-scale images, outperforms the baselines by a large margin over all of the compared evaluation metrics. Especially, it represents an improvement of 1.8 points for the AP when compared to the original module.

Table 1. Comparisons of **single model** results on the MS COCO object detection benchmark.

	AP	**AP$_{@0.5}$**	**AP$_{@0.75}$**	**AP$_S$**	**AP$_M$**	**AP$_L$**	**Backbone**
YOLOv3 [42] + FPN	33.0	57.9	34.4	18.3	35.4	41.9	DarkNet-53 [43]
RentinaNet [44] + FPN	39.1	59.1	42.3	21.8	42.7	50.2	ResNet-101
Faster R-CNN + FPN	36.2	59.1	39.0	18.2	39.0	48.2	ResNet-101
Faster R-CNN + G-RMI [45]	34.7	55.5	36.7	13.5	38.1	52.0	Inception-ResNet-v2 [46]
Faster R-CNN + TDM [47]	36.8	57.7	39.2	16.2	39.8	52.1	Inception-ResNet-v2
Mask R-CNN + FPN	38.2	60.3	41.7	20.1	41.1	50.2	ResNet-101
Mask R-CNN + FPN	39.8	62.3	43.4	22.1	43.2	51.2	ResNeXt-101 [48]
Mask R-CNN + APN	**41.0**	**63.6**	**44.3**	**22.8**	**44.1**	**52.7**	ResNet-50

In Table 1, when the FPN is applied to the Mask R-CNN, the best effect is achieved in the related baseline methods. The ResNet-100 is used by the network as the backbone network, and ResNet-100 can usually achieve better recognition than ResNet-50 [21]. The backbone network of Our APN is ResNet-50, but it still achieved a better effect than the FPN applied to the Mask R-CNN, which uses ResNet-100 as the backbone network. Therefore, our APN network is more effective than FPN in object recognition. In addition, we plotted the performance comparison curves for the APN and baseline methods. In Figure 4, the abscissa is the threshold of IoU, and the ordinate is the AP value of the APN and baseline methods. We use different color polylines to distinguish different methods. Obviously, we can see that the APN proposed by us has higher AP values than any other methods in any threshold range.

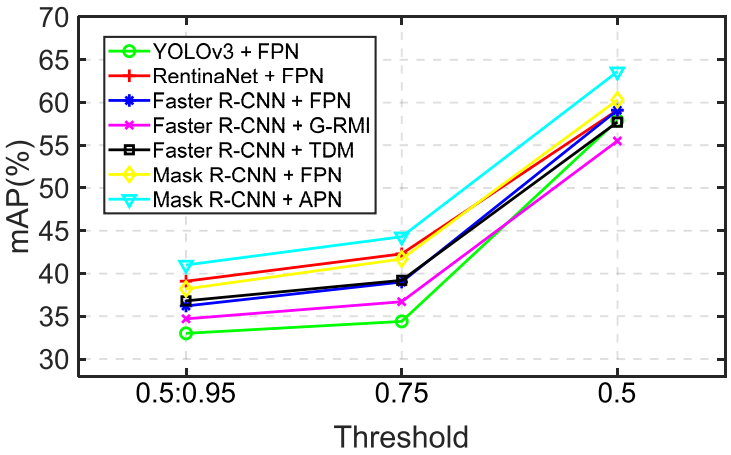


Figure 4. Mean Average Precision of the proposed APN-based model and FPN-based models in different threshold.

Figure 5 shows the comparison of the methods based on our model and the FPN. According to the results, we can see that our improvement model detects more qualitative objects, which means our model can detect those objects that highly overlap, but that belong to different individuals. The design of our models takes advantage of the contextual information from the whole feature pyramid to fulfill the object detection.

Figure 5. Selected examples from the MS COCO 2017 `test` dataset using our method and other methods using the FPN model.

Then, we performed the instance segmentation experiments on the COCO dataset. We report the performance of the proposed APN on the COCO dataset for comparison. As shown in Table 2, the APN applied to Mask R-CNN trained and tested on single-scale images already outperforms the FPN applied to Mask R-CNN, and both use ResNet-50 as their backbone network. All instantiations of our proposed model outperform the base variants of the latter model by nearly 2.0 points on average.

Table 2. The value of instance segmentation results for the COCO dataset. All entries are single-model results.

	AP^M	$AP^M_{@0.5}$	$AP^M_{@0.75}$	AP^M_S	AP^M_M	AP^M_L
Mask R-CNN + FPN	46.5	72.4	51.0	29.2	50.6	59.7
Mask R-CNN + APN	48.3	74.7	53.1	31.2	52.4	61.4
	+1.8	+2.3	+2.1	+2.0	+2.4	1.7

Moreover, Figure 6 shows the partial examples of the segmentation results using the APN. In Figure 6, we can see that different classes of objects can be accurately segmented.

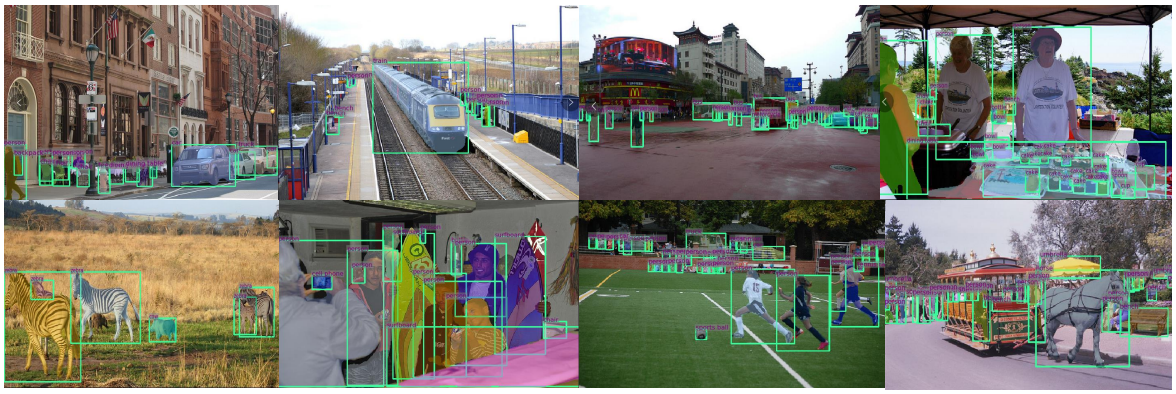


Figure 6. Selected examples of the test results on the COCO `test-dev` images, using the proposed APN and running at 5 fps.

In Table 3, we compare two different designs of our fusion block. Our simple channel-wise addition fusion design results in a severe decrease in bounding box loss AP (2.4 points). Compared with the simple channel-wise addition fusion strategy, the convolutional fusion strategy takes all pixels from the feature maps into the calculation. This suggests that the more the spatial information is considered in the fusion block, the better are the results the fusion block gains. The result was tested on the MS COCO `val-2017` set, and the backbone network was ResNet-50. Thus, adding a convolutional layer to fuse the reweighted feature map obtains greater gains over simple channel-wise fusion.

Table 3. The test results of the APN when the channel-wise addition fusion and convolutional fusion are used.

	AP	**AP$_{@0.5}$**	**AP$_{@0.75}$**	**AP$_S$**	**AP$_M$**	**AP$_L$**
Mask R-CNN + APN(channel-wise addition fusion)	53.0	76.2	59.9	35.9	58.0	64.1
Mask R-CNN + APN(convolutional fusion)	55.4	78.5	63.2	38.7	60.6	66.4
	+2.4	+2.3	+3.3	+2.8	+1.4	+2.3

We further tested the efficiency of the APN for the task of object detection on the COCO set `test-dev`. The comparison in Table 4 reveals that, with much fewer parameters, the APN compared with the FPN when the backbone is ResNet-101 can still achieve better performance. Based on the same backbone of ResNet-50, the proposed APN can achieve much better detection results. As shown in Table 4, the APN based on the ResNet-50 is 2.8 points better than the FPN based on the ResNet-101 with nearly 20 million fewer parameters and the smaller backbone of ResNet-50. As the performance of ResNeXt-101 is better than ResNet-50, and with a larger backbone model, the performance of the APN can achieve better results. To test the efficiency of our design, we carried out more ablation studies of the individual module of the APN. Figure 2 shows that the feature attention fusion block we proposed is composed of the adaptive transformation module and feature attention block and fusion block. Next, we tested the detection performance of the small module in the feature attention fusion block when it is assembled separately and multiplied. In Table 4, we can see that the third row is the case where the APN only contains the adaptive transformation module, and the fourth row the simultaneous existence of the adaptive transformation module and the feature attention block. The fusion block used in these two cases is channel-wise addition fusion. The fusion block used by the APN in the last row of the table is convolutional fusion, which means that this APN used a complete feature attention fusion block structure.

Table 4. Ablation experiments: We trained on `trainval35k`, tested on `val-2017` or `test-dev`, and report the value of the AP.

	AP	Million Parameters	Backbone
Mask R-CNN + FPN	37.9	84.3	ResNet-50
Mask R-CNN + FPN	38.2	184.7	ResNet-101
Mask R-CNN + APN(only adaptive transformation)	38.0(+0.1)	90.1(+5.8)	ResNet-50
Mask R-CNN + APN(adaptive transformation+attention block)	40.3(+2.4)	136.7(+52.4)	ResNet-50
Mask R-CNN + APN	41.0(+3.1)	158.9(+74.6)	ResNet-50

As shown in Table 4, when the APN only contains the adaptive transformation module, the AP value of APN is 0.1 higher than that of the FPN using Resnet-50 as the backbone network. Thus, the adaptive transformation of feature maps from all stages marginally improves performance. However, from the channel-wise addition fusion strategy and the conv fusion on the last two rows of Table 4, we can see that the AP value of the APN on the last row of the table using the conv fusion strategy is 0.7 more than the APN of the fourth row of the table using the channel-wise addition fusion. Therefore, the conv fusion strategy gains greater benefits from fusing feature maps than the channel-wise addition fusion strategy. In addition, from the third and fourth rows in Table 4, we can see that, when the APN uses the feature attention block, the AP value is 2.3 more than when the APN only contains the adaptive transformation module. Rather than simply giving all the channels of the feature map the same weight as the FPN, our feature attention block learns to reweight different channels from all network stages. Such a reweight-strategy helps the network learn, through the training dataset, to take advantage of feature maps from different network stages to construct a better fusion feature map.

The APN utilizes both the multiscale feature fusion and the channel-wise attention mechanism to generate new feature pyramids. It can be proven that the FPN is a special case of the APN. We look forward to seeing more application to replace our proposed APN with the FPN and yield better performance. In addition, we also carried out an experiment to analyze the inference time of the proposed module. The proposed APN ran in 4 ms on average during the inference time, which was ignorable compared with the inference time of the backbone ConvNet. The experiment was performed with the ResNet-50 model on a single P40 where the input size of the image was 512×512.

4.3. Experiment on the 2018 WAD Dataset

We followed the standard evaluation metrics: the mAP, i.e., AP, $AP_{@0.5}$, $AP_{@0.75}$, AP_S, AP_M, and AP_L. Specifically, our average precision was calculated based on the top 100 proposals. We also used two images in one image batch for the training. Because the dataset is full of images with high resolutions, we chose the minimum and maximum resolutions of the resized images to be 800 and 1024. We adopted stochastic gradient descent training on two NVidia TitanX GPUs with a batch size of one. The initial learning rate was set to 0.001, and it was decreased by a factor of 0.1 after 100 k and 120 k iterations and finally terminated at 140 k iterations. We did not adopt any augmentation for the training, which we found made it harder to converge.

Owing to the small size of the mini-batch, we chose to freeze the batch normalization layer. The learning rate was different from the Mask R-CNN and the FPN because of the different platforms. Other implementation details were the same as He et al. [18], and models were trained on the training subset and used ResNet-101 [21]. Moreover, Figure 7 shows part of the results of object detection.

Table 5b shows the object detection results for the comparison of our model to the baseline FPN. We present the results with ResNet-50 pre-trained on the COCO dataset, which was also trained and tested on single-scale images. According to the results, our model outperforms the baseline by a large margin over all of the compared evaluation metrics. This represents an improvement of 1.7 points for *AP* compared with the original module. To specify the effectiveness of our module, we divided the test-dev into separate classes. Table 5a presents the results for different separated classes in the `test-dev` dataset. Note that the results of the APN show better results across the detection results of all sizes, and achieves nearly two points over all classes and all sizes on average. The performance of

the separated classes also improves significantly, which indicates the benefit of using a fusion module with contextual information. Therefore, the APN proposed by us is better than the FPN in target recognition of different categories or the whole.

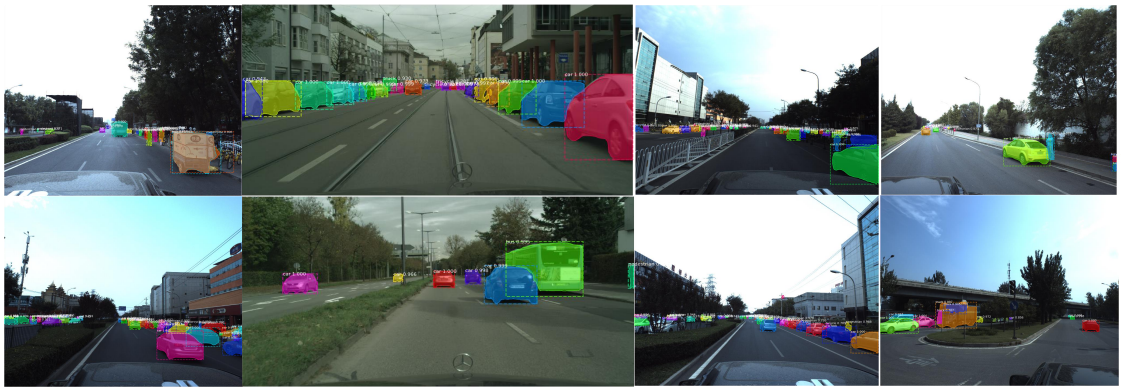


Figure 7. Images in each row are the object detection and instance segmentation results.

Table 5. Object detection results using Mask R-CNN [18] evaluated on the 2018 WAD `test-dev` set: (**a**) object detection results on separate class of 2018 WAD `test-dev`; and (**b**) object detection results of 2018 WAD `test-dev`.

(a)						
	AP	**AP$_{@0.5}$**	**AP$_{@0.75}$**	**AP$_S$**	**AP$_M$**	**AP$_L$**
FPN$_{car}$	35.7	59.8	36.8	11.7	48.1	77.5
APN$_{car}$	39.8	61.3	42.4	15.7	53.5	79.2
FPN$_{moto}$	23.5	46.8	20.3	7.1	26.2	47.1
APN$_{moto}$	28.1	51.6	27.9	10.0	31.3	52.0
FPN$_{bike}$	12.6	30.5	8.2	1.6	14.5	31.1
APN$_{bike}$	15.7	34.0	11.1	3.5	17.5	36.2
FPN$_{person}$	16.4	36.2	12.1	4.9	31.6	58.2
APN$_{person}$	18.3	38.3	14.7	6.3	35.3	57.0
FPN$_{truck}$	44.7	69.3	50.6	17.0	50.1	65.4
APN$_{truck}$	48.8	73.4	58.1	21.9	54.9	66.0
FPN$_{bus}$	41.0	66.7	44.3	10.7	47.4	67.2
APN$_{bus}$	46.0	71.7	51.3	17.3	56.4	68.8
FPN$_{tricycle}$	34.7	61.6	36.4	29.2	36.2	47.7
APN$_{tricycle}$	37.3	65.3	40.1	31.9	37.8	50.9
(b)						
	AP	**AP$_{@0.5}$**	**AP$_{@0.75}$**	**AP$_S$**	**AP$_M$**	**AP$_L$**
FPN$_{all}$	30.7	49.7	29.2	12.8	34.4	55.7
APN$_{all}$	32.4	54.7	33.8	15.4	39.4	59.2
	+1.7	+5.0	+4.6	+2.6	+5.0	+3.5

Similar to the object detection results on the Mask R-CNN, we also report the instance segmentation performance of our proposed module on the WAD test-dev for comparison. As shown in Table 6, compared with the baseline, our APN consistently improves the performance across different evaluation metrics.

The result of the segmentation on the evaluation metrics shows a significant improvement of 1.3 points for AP compared with the original FPN. In Figure 7, we can see the results of instance segmentation.

Table 6. Instance segmentation proposals evaluated on the first 1 k 2018 WAD `test-dev` images, denoted as AP^M. All models were trained on the `train` set.

	AP^M	$AP^M_{@0.5}$	$AP^M_{@0.75}$	AP^M_S	AP^M_M	AP^M_L
FPN_{all}	38.2	46.9	28.1	11.7	32.8	53.7
APN_{all}	39.5	51.7	30.0	12.7	36.3	55.1
	+1.3	+4.8	+1.9	+1.0	+3.5	+1.4

5. Conclusions

In this paper, we propose the APN, which utilizes both the multiscale feature fusion and the multiscale-aware channel-wise attention mechanism to generate new feature pyramids for the task of object detection. We reshape the features from all feature levels and shorten the distance between the lower and topmost feature levels to enable reliable information propagation. The APN is a lightweight network and the efficient feature attention block can enhance important feature channels. In addition, the APN adopts the feature fusion strategy of conv and channel-wise addition, which will help to improve the recognition performance of the whole network.The running time of APN is 4 ms, which can be ignored compared with backbone network. Experiments on the MS COCO and the 2018 WAD datasets proved that the APN significantly improves the performance compared with the contrast methods, and without any bells and whistles. In the future, we will consider the utilization of video and RGB-D data for the task of object detection and extend our method to support more general detection models.

Author Contributions: Conceptualization, J.Z. and W.W.; Formal analysis, J.Z. and Z.C.; Investigation, J.Z. and Z.C.; Methodology, J.Z.; Project administration, W.W.; Validation, Y.Y.; Writing—original draft, J.Z. and Y.Y.; and Writing—review and editing, J.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yan, X.; Cui, B.; Xu, Y.; Shi, P.; Wang, Z. A Method of Information Protection for Collaborative Deep Learning under GAN Model Attack. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2019**. [CrossRef] [PubMed]
2. Sugimori, H.; Kawakami, M. Automatic Detection of a Standard Line for Brain Magnetic Resonance Imaging Using Deep Learning. *Appl. Sci.* **2019**, *9*, 3849. [CrossRef]
3. Xu, Y.; Ren, J.; Wang, G.; Zhang, C.; Yang, J.; Zhang, Y. A Blockchain-based Nonrepudiation Network Computing Service Scheme for Industrial IoT. *IEEE Trans. Industr. Inform.* **2019**, *15*, 3632–3641. [CrossRef]
4. Xu, Y.; Ren, J.; Zhang, Y.; Zhang, C.; Shen, B.; Zhang, Y. Blockchain Empowered Arbitrable Data Auditing Scheme for Network Storage as a Service. *IEEE Trans. Serv. Comput.* **2019**. [CrossRef]
5. Xia, G.S.; Bai, X.; Ding, J.; Zhu, Z.; Belongie, S.; Luo, J.; Datcu, M.; Pelillo, M.; Zhang, L. DOTA: A large-scale dataset for object detection in aerial images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 3974–3983.
6. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
7. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
8. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; ACM: New York, NY, USA, 2015; pp. 91–99.
9. Wang, G.; Guo, J.; Chen, Y.; Li, Y.; Xu, Q. A PSO and BFO-Based Learning Strategy Applied to Faster R-CNN for Object Detection in Autonomous Driving. *IEEE Access* **2019**, *7*, 18840–18859. [CrossRef]

10. Dai, J.; Li, Y.; He, K.; Sun, J. R-FCN: Object detection via region-based fully convolutional networks. In Proceedings of the Advances in Neural Information Processing Systems, (NIPS 2016), Barcelona, Spain, 5–10 December 2016; ACM: New York, NY, USA, 2016; pp. 379–387.
11. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37.
12. Cai, Z.; Fan, Q.; Feris, R.S.; Vasconcelos, N. A unified multi-scale deep convolutional neural network for fast object detection. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 354–370.
13. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Araucano Park, Chile, 11–18 December 2015; pp. 1440–1448.
14. Xie, S.; Tu, Z. Holistically-nested edge detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1395–1403.
15. Wang, X.; Ma, H.; Chen, X.; You, S. Edge preserving and multi-scale contextual neural network for salient object detection. *IEEE Trans. Image Process.* **2018**, *27*, 121–134. [CrossRef] [PubMed]
16. Lin, T.Y.; Dollár, P.; Girshick, R.; Kaiming, H.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
17. Fu, C.Y.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. DSSD: Deconvolutional single shot detector. *arXiv* **2017**, arXiv:1701.06659.
18. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
19. Lee, G.; Tai, Y.W.; Kim, J. Deep saliency with encoded low level distance map and high level features. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 660–668.
20. Fu, J.; Zheng, H.; Mei, T. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4438–4446.
21. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
22. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Berlin/Heidelberg, Germany, 2014; pp. 740–755.
23. Mangai, U.G.; Samanta, S.; Das, S.; Chowdhury, P.R. A survey of decision fusion and feature fusion strategies for pattern classification. *IETE Tech. Rev.* **2010**, *27*, 293–307. [CrossRef]
24. Lan, X.; Ma, A.J.; Yuen, P.C. Multi-cue visual tracking using robust feature-level fusion based on joint sparse representation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1194–1201.
25. Lan, X.; Ma, A.J.; Yuen, P.C.; Chellappa, R. Joint sparse representation and robust feature-level fusion for multi-cue visual tracking. *IEEE Trans. Image Process.* **2015**, *24*, 5826–5841. [CrossRef] [PubMed]
26. Song, W.; Li, S.; Fang, L.; Lu, T. Hyperspectral image classification with deep feature fusion network. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 3173–3184. [CrossRef]
27. Long, H.; Chung, Y.; Liu, Z.; Bu, S. Object Detection in Aerial Images Using Feature Fusion Deep Networks. *IEEE Access* **2019**, *7*, 30980–30990. [CrossRef]
28. Tian, K.; Zeng, L.; McGrath, S.; Yin, Q.; Wang, W. 3D Facial Expression Recognition Using Deep Feature Fusion CNN. In Proceedings of the 2019 30th Irish Signals and Systems Conference (ISSC), Maynooth, Ireland, 17–18 June 2019; pp. 1–6.
29. Starzacher, A.; Rinner, B. Embedded realtime feature fusion based on ANN, SVM and NBC. In Proceedings of the 2009 12th International Conference on Information Fusion, Seattle, WA, USA, 6–9 July 2009; pp. 482–489.
30. Luong, T.; Pham, H.; Manning, C.D. Effective Approaches to Attention-based Neural Machine Translation. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 1412–1421.

31. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; ACM: New York, NY, USA, 2017; pp. 5998–6008.
32. Wang, F.; Jiang, M.; Qian, C.; Yang, S.; Li, C.; Zhang, H.; Wang, X.; Tang, X. Residual attention network for image classification. *arXiv* **2017**, arXiv:1704.06904.
33. Fan, Q.; Zhuo, W.; Tai, Y.W. Few-Shot Object Detection with Attention-RPN and Multi-Relation Detector. *arXiv* **2019**, arXiv:1908.01998.
34. Wang, X.; Cai, Z.; Gao, D.; Vasconcelos, N. Towards universal object detection by domain attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 7289–7298.
35. Zhang, C.; Kim, J. Object Detection With Location-Aware Deformable Convolution and Backward Attention Filtering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 9452–9461.
36. Li, Y.L.; Wang, S. HAR-Net: Joint Learning of Hybrid Attention for Single-stage Object Detection. *arXiv* **2019**, arXiv:1904.11141.
37. Huang, Z.; Ke, W.; Huang, D. Improving Object Detection with Inverted Attention. *arXiv* **2019**, arXiv:1903.12255.
38. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv* **2016**, arXiv:1602.07360.
39. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 1492–1500.
40. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A System for Large-Scale Machine Learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
41. Girshick, R. Fast R-CNN. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1440–1448.
42. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
43. Redmon, J.; Farhadi, A. YOLO9000: better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
44. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence, Venice, Italy, 22–29 October 2018; pp. 2980–2988.
45. Huang, J.; Rathod, V.; Sun, C.; Zhu, M.; Korattikara, A.; Fathi, A.; Fischer, I.; Wojna, Z.; Song, Y.; Guadarrama, S.; et al. Speed/accuracy trade-offs for modern convolutional object detectors. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7310–7311.
46. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–10 February 2017; ACM: New York, NY, USA, 2017; pp. 500–515.
47. Shrivastava, A.; Sukthankar, R.; Malik, J.; Gupta, A. Beyond skip connections: Top-down modulation for object detection. *arXiv* **2016**, arXiv:1612.06851.
48. Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; He, K. Aggregated residual transformations for deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1492–1500.