Rethinking Classification and Localization for Object Detection

Yue Wu[1], Yinpeng Chen[2], Lu Yuan[2], Zicheng Liu[2], Lijuan Wang[2], Hongzhi Li[2] and Yun Fu[1]
[1]Northeastern University, [2]Microsoft

Abstract

Two head structures (i.e. fully connected head and convolution head) have been widely used in R-CNN based detectors for classification and localization tasks. However, there is a lack of ***understanding*** *of how does these two head structures work for these two tasks. To address this issue, we perform a thorough analysis and find an interesting fact that the two head structures have* ***opposite*** *preferences towards the two tasks. Specifically, the fully connected head (fc-head) is more suitable for the classification task, while the convolution head (conv-head) is more suitable for the localization task. Furthermore, we examine the weight matrix in the fc-head and find that it learns spatial sensitive transformations. We believe that this allows fc-head to distinguish a complete object from part of an object, but is not robust to regress the whole object. Based upon these findings, we propose a* ***Double-Head*** *method, which has a fully connected head focusing on classification and a convolution head for bounding box regression. Without bells and whistles, our method gains +3.5 and +2.8 AP on MS COCO dataset from Feature Pyramid Network (FPN) baselines with ResNet-50 and ResNet-101 backbones, respectively.*

1. Introduction

Most two-stage object detectors [10, 11, 34, 4, 26] share a head for both classification and bounding box regression. Two different head structures are widely used. Faster R-CNN [34] uses a convolution head (conv5) on a single level feature map (conv4), while FPN [26] uses a fully connected head (2-*fc*) on multiple level feature maps. However, there is a lack of **understanding** between these two head structures with respect to the two tasks (object classification and localization).

In this paper, we perform a thorough comparison between the fully connected head (*fc-head*) and the convolution head (*conv-head*) on the two detection tasks, i.e. object classification and localization. We find that *these two different head structures are complementary*. *fc-head* is more suitable for the classification task as its classification score is more correlated to intersection over union (IoU) between a proposal and the ground truth box. Meanwhile, *conv-head* provides more accurate bounding box regression.

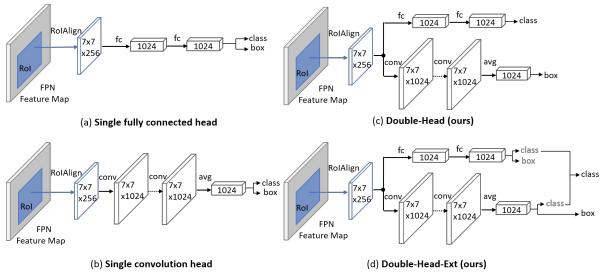

Figure 1. Comparison between single head and double heads, (a) single fully connected (2-*fc*) head, (b) single convolution head, (c) Double-Head method, which splits classification and localization on a fully connected head and a convolution head respectively, and (d) Double-Head-Ext, which extends Double-Head by introducing supervision from unfocused tasks during training and combining classification scores from both heads during inference.

We believe this is because *fc-head* is spatial sensitive, having different parameters for different parts of a proposal, while *conv-head* shares convolution kernels for all parts. To validate this, we examine the weight matrix for the first fully connected layer in *fc-head* and confirm its spatial sensitivity. The spatial sensitivity enables *fc-head* to distinguish between a complete object and part of an object (classification), but is not robust to determine the offset of the whole object (bouding box regression).

In light of above findings, we propose a Double-Head method, which includes a fully connected head (*fc-head*) for classification and a convolution head (*conv-head*) for bounding box regression (see Figure 1-(c)), to leverage advantages of both heads. This design outperforms both single *fc-head* and single *conv-head* (see Figure 1-(a), (b)) by a non-negligible margin. In addition, we extend Double-Head (Figure 1-(d)) to further improve the accuracy by leveraging unfocused tasks (i.e. classification in *conv-head*, and bounding box regression in *fc-head*). Our method outperforms FPN baseline by a non-negligible margin on MS COCO dataset, gaining 3.5 and 2.8 AP for using ResNet-50 and ResNet-101 backbones, respectively.

2. Related Work

One-stage Object Detectors: OverFeat [36] detects objects by sliding windows on feature maps. SSD [29, 9] and YOLO [31, 32, 33] have been tuned for speed by predicting object classes and locations directly. RetinaNet [27] alleviates the extreme foreground-background class imbalance problem by introducing focal loss. Point-based methods [21, 22, 46, 7, 47] model an object as keypoints (corner, center, etc), and are built on keypoint estimation networks.

Two-stage Object Detectors: RCNN [12] applies a deep neural network to extract features for proposals generated by selective search [41]. SPPNet [14] speeds up RCNN significantly using spatial pyramid pooling. Fast RCNN [10] improves the speed and performance utilizing a differentiable RoI Pooling. Faster RCNN [34] introduces Region Proposal Network (RPN) to generate proposals. R-FCN [4] employs position sensitive RoI pooling to address the translation-variance problem. FPN [26] builds a top-down architecture with lateral connections to extract features across multiple layers.

Backbone Networks: Fast RCNN [10] and Faster RCNN [34] extract features from conv4 of VGG-16 [37], while FPN [26] utilizes features from multiple layers (conv2 to conv5) of ResNet [15]. Deformable ConvNets [5, 48] propose deformable convolution and deformable Region of Interests (RoI) pooling to augment spatial sampling locations. Trident Network [24] generates scale-aware feature maps with multi-branch architecture. MobileNet [17, 35] and ShuffleNet [45, 30] introduce efficient operators (like depth-wise convolution, group convolution, channel shuffle, etc) to speed up on mobile devices.

Detection Heads: Light-Head RCNN [25] introduces an efficient head network with thin feature maps. Cascade RCNN [3] constructs a sequence of detection heads trained with increasing IoU thresholds. Feature Sharing Cascade RCNN [23] utilizes feature sharing to ensemble multi-stage outputs from cascade RCNN [3] to improve the results. Mask RCNN [13] introduces an extra head for instance segmentation. COCO Detection 18 Challenge winner (Megvii) [1] coupes bounding box regression and instance segmentation in the convolution head. IoU-Net [20] introduces a branch to predict IoUs between detected bounding boxes and ground truth boxes. Similar to IoU-Net, Mask Scoring RCNN [18] presents an extra head to predict Mask IoU scores for each segmentation mask. He et. al. [16] learns uncertainties of bounding box prediction with an extra task to improve the localization results. Learning-to-Rank [38] utilizes an extra head to produce a rank value of a proposal for Non-Maximum Suppression (NMS). Zhang and Wang [44] point out that there exist mis-alignments between classification and localization task domains. In contrast to existing methods, which apply a head to extract Region of Interests (RoI) features for both classification and bounding box regression tasks, we propose to split these two tasks into different heads, based upon our thorough analysis and understanding of detection heads with the two tasks.

3. Analysis: Comparison between *fc-head* and *conv-head*.

In this section, we compare *fc-head* and *conv-head* for both classification and bounding box regression. For each head, we train a model with FPN backbone [26] using ResNet-50 [15] on MS COCO 2017 dataset [28]. The *fc-head* includes two fully connected layers. The *conv-head* has five residual blocks. The evaluation and analysis is conduct on the *val* with 5,000 images. *fc-head* and *conv-head* have 36.8% and 35.9% AP, respectively.

3.1. Data Processing for Analysis

To make a fair comparison, we perform analysis for both heads on predefined proposals rather than proposals generated by RPN [34], as the two detectors have different proposals. The predefined proposals include sliding windows around the ground truth box with different sizes. For each ground truth object, we generate about 14,000 proposals, whose IoUs with the ground truth box gradually change from zero (background) to one (the ground truth box). For each proposal, both detectors (*fc-head* and *conv-head*) generate classification scores and regressed bounding boxes. We apply this process for all objects in the *val*.

To calculate the statistics, we split these predefined proposals into 20 bins, by uniformly splitting the proposal IoUs within the range $[0, 1]$. For each bin, we calculate mean and standard deviation of classification scores and IoUs of regressed boxes. In order to be more informative, we split all objects in the *val* into three groups by two criteria: the size of the object and the difficulty of the object category. The grouping based on object size (small/medium/large) strictly follows the standard COCO evaluation. Regarding the difficulty, we rank all object classes based upon AP results of the FPN [26] baseline. Each group on difficulty of the object category (easy/medium/hard) takes one third of classes. The results are shown in Figure 2.

3.2. Comparison on Classification Task

We plot classification scores for both *fc-head* and *conv-head* in the first row of Figure 2. Compared to *conv-head*, *fc-head* provides higher scores for high quality proposals (with higher IoU). Furthermore, the mean ranges (difference between classification scores at high IoUs and low IoUs) enlarge from large objects to small objects and from easy classes to hard classes. Larger mean ranges from *fc-head* indicate that *classification scores of fc-head are more correlated to proposals IoUs than of conv-head*, especially for small objects and hard classes. To validate this, we

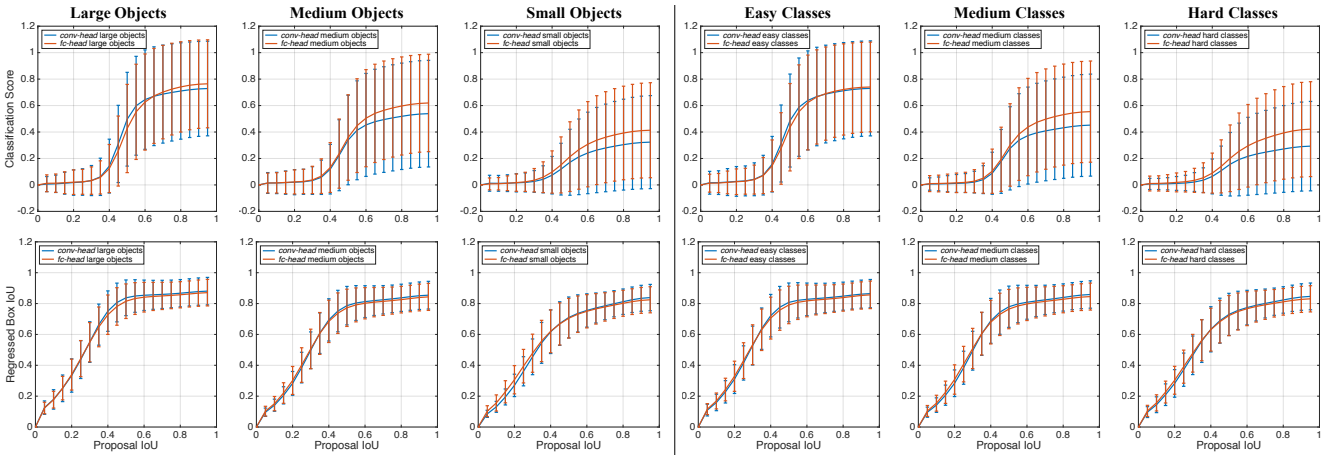

Figure 2. Comparison of classification and bounding box regression between *fc-head* and *conv-head*. Mean and standard deviation of classification scores and regressed box IoUs at different proposal IoUs are in the first row and second row, respectively. Different sizes and difficulty of classes are in the left and right, respectively. Classification scores in *fc-head* are more correlated to proposals IoUs than in *conv-head*. *conv-head* has better regression results than *fc-head*. (Best viewed in color)

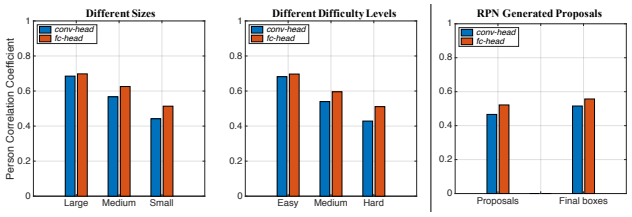

Figure 3. Pearson Correlation Coefficient between classification scores and boxes IoUs for different sizes and different difficulty of classes with sliding window proposals in the left, for proposals generated by RPN and final boxes in the right.

compute the Pearson Correlation Coefficient to measure the correlation between proposal IoUs and classification scores. The results (shown in Figure 3) demonstrate that the classification scores of *fc-head* are more correlated to the proposal IoUs.

To confirm the above finding in real scenarios, we compute Pearson Correlation Coefficients for the real proposals generated by RPN [34]. We compute the correlation between classification scores and bounding box IoUs for (a) all proposals, and (b) final detected boxes after NMS (shown in Figure 3). Similarly, classification scores of *fc-head* are more correlated to bounding box IoUs than *conv-head* for both cases. This property makes sense as detected boxes with higher IoUs should have higher classification scores so that these boxes can be ranked high in order, which is also friendly to the evaluation measurement (AP).

3.3. Comparison on Localization Task

We plot regressed box IoUs for both *fc-head* and *conv-head* in the second row of Figure 2. Compared to *fc-head*, *conv-head* has higher regressed box IoUs with high proposal IoUs. This demonstrates that *conv-head has better regression ability than fc-head*.

3.4. Discussion

Why does fc-head show more correlation between the classification scores and proposal IoUs, and perform worse in localization? We believe that it is because *fc-head* is more spatial sensitive than *conv-head*. Intuitively, *fc-head* applies *unshared* transformation (fully connected layer) on every position of input feature maps and implicitly embeds the spatial information in the representation for the proposal. This enables *fc-head* to distinguish between a complete object and part of an object, but is not robust to determine the offset of the whole object. In contrast, *conv-head* uses *shared* transformation (convolutional kernels) on every position of input feature maps, then uses average pooling to aggregate and finally generates the representation for a proposal.

To validate the spatial sensitivity, we treat a detection head as a transformation from region-specific feature maps (output of RoIAlign with dimension $C_{in} \times 7 \times 7$) to the global representation of a proposal (a vector with dimension C_{out}). And, we design a correlation analysis scheme to examine if the transformations at different positions in the 7×7 region are correlated. First, each cell of 7×7 grid in the feature map is corresponding to a $C_{in} \times C_{out}$ weight. For each pair of cells, we compute the correlation between their weights, resulting a 49×49 correlation matrix (one row per cell). Then we convert each row to a 7×7 matrix to show the correlation between a cell with all other cells by keeping their 2D relationship. Figure 4 shows the correlation analysis results of the *fc-head*, where the trans-

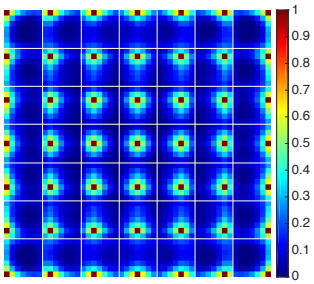

Figure 4. Pairwise relationship between weight parameters in the first fully connected layer in *fc-head*. Each cell represents one weight parameter of the corresponding position in the 7×7 feature maps (output of RoIAlign). At each cell, a 7×7 matrix shows the correlation between the cell with all other cells by keeping their 2D relationship.

formation tensor can be found in the first fully connected layer. We can see that each cell is only correlated to its nearest neighbors. Therefore *fc-head* is spatial sensitive and can embed spatial sensitive features in the final global representation for the proposal, making it easier to discriminate if one proposal covers one complete or partial object. On the other hand, it is not as robust as *conv-head* to regress the bounding box.

4. Our Approach: Double-Head

Given above analysis, it is neutrally to propose a double-head method to take advantage of the different merits from different head structures. In this section, we discuss our Double-Head method in detail. Firstly, we discuss the network structure of Double-Head (Figure 1-(c)), which has a fully connected head (*fc-head*) for classification and a convolution head (*conv-head*) for bounding box regression. Then, we extend Double-Head to Double-Head-Ext (Figure 1-(d)) by leveraging unfocused tasks (i.e. bounding box regression in *fc-head* and classification in *conv-head*).

4.1. Network Structure

Our Double-Head method (see Figure 1-(c)) splits classification and localization into *fc-head* and *conv-head*, respectively. The details of backbone and head networks are described as follows:

Backbone: We use FPN [26] backbone to generate region proposals and extract object features from multiple levels using RoIAlign [13]. Each proposal has a feature map with size $256 \times 7 \times 7$, which is transformed by *fc-head* and *conv-head* into two feature vectors (each with dimension 1024) for classification and bounding box regression, respectively.

Fully Connected Head (*fc-head*) has two fully connected layers (see Figure 1-(c)), following the design in FPN [26] (Figure 1-(a)). The output dimension is 1024. The parameter size is 13.25M.

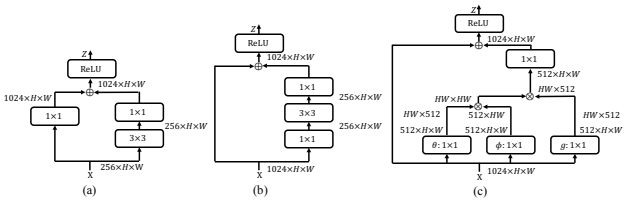

Figure 5. Network architectures of three components: (a) residual block to increase the number of channels (from 256 to 1024), (b) residual bottleneck block, and (c) non-local block.

Convolution Head (*conv-head*) stacks K residual blocks [15]. The first block increases the number of channels from 256 to 1024 (shown in Figure 5-(a)), and others are bottleneck blocks [15] (shown in Figure 5-(b)). At the end, average pooling is used to generate the feature vector with dimension 1024. Each residual block has 1.06M parameters. We also introduce a variation for the convolution head by inserting a non-local block [42] (see Figure 5-(c)) before each bottleneck block to enhance foreground objects. Each non-local block has 2M parameters.

Loss Function: Both heads (i.e. *fc-head* and *conv-head*) are jointly trained with region proposal network (RPN) end to end. The overall loss is computed as follows:

$$\mathcal{L} = \omega^{fc}\mathcal{L}^{fc} + \omega^{conv}\mathcal{L}^{conv} + \mathcal{L}^{rpn}, \tag{1}$$

where ω^{fc} and ω^{conv} are weights for *fc-head* and *conv-head*, respectively. $\mathcal{L}^{fc}$, $\mathcal{L}^{conv}$, $\mathcal{L}^{rpn}$ are the losses for *fc-head*, *conv-head* and RPN, respectively.

4.2. Extension: Leveraging Unfocused Tasks

In vanilla Double-Head, each head focuses on its assigned task (i.e. classification in *fc-head* and bounding box regression in *conv-head*). In addition, we found that unfocused tasks (i.e. bounding box regression in *fc-head* and classification in *conv-head*) are helpful in two aspects: (a) bounding box regression provides auxiliary supervision for *fc-head*, and (b) classifiers from both heads are complementary. Therefore, we introduce unfocused task supervision in training and propose a complementary fusion method to combine classification scores from both heads during inference (see Figure 1-(d)). This extension is referred to Double-Head-Ext.

Unfocused Task Supervision: Due to the introduction of the unfocused tasks, the loss for *fc-head* ($\mathcal{L}^{fc}$) includes both classification loss and bounding box regression loss as follows:

$$\mathcal{L}^{fc} = \lambda^{fc} L^{fc}_{cls} + (1 - \lambda^{fc}) L^{fc}_{reg}, \tag{2}$$

where L^{fc}_{cls} and L^{fc}_{reg} are the classification and bounding box regression losses in *fc-head*, respectively. λ^{fc} is the weight

that controls the balance between the two losses in *fc-head*. In the similar manner, we define the loss for the convolution head ($\mathcal{L}^{conv}$) as follows:

$$\mathcal{L}^{conv} = (1 - \lambda^{conv})L_{cls}^{conv} + \lambda^{conv} L_{reg}^{conv}, \quad (3)$$

where L_{cls}^{conv} and L_{reg}^{conv} are classification and bounding box regression losses in *conv-head*, respectively. Different with λ^{fc}, the balance weight λ^{conv} is multiplied by the regression loss L_{reg}^{conv}, as the bounding box regression is the focused task in *conv-head*. Note that the vanilla Double-Head is a special case when $\lambda^{fc} = 1$ and $\lambda^{conv} = 1$. Similar to FPN [26], cross entropy loss is applied to classification, and Smooth-L_1 loss is used for bounding box regression.

Complementary Fusion of Classifiers: We believe that the two heads (i.e. *fc-head* and *conv-head*) capture complementary information for object classification due to their different structures. Therefore we propose to fuse the two classifiers as follows:

$$s = s^{fc} + s^{conv}(1 - s^{fc}) = s^{conv} + s^{fc}(1 - s^{conv}), \quad (4)$$

where s^{fc} and s^{conv} are classification scores from *fc-head* and *conv-head*, respectively. The increment from the first score (e.g. s^{fc}) is a product of the second score and the reverse of the first score (e.g. $s^{conv}(1 - s^{fc})$). This is different from [3] which combining all classifiers by average. Note that this fusion is only applicable when $\lambda^{fc} \neq 0$ and $\lambda^{conv} \neq 1$.

5. Experimental Results

We evaluate our approach on MS COCO 2017 dataset [28] and Pascal VOC07 dataset [8]. MS COCO 2017 dataset has 80 object categories. Training is conduct on *train* with 118K images and the evaluation is on *val* with 5K images and *test-dev* with 41K images. Object detection accuracy is measured by the standard COCO-style Average Precision (AP) with different IoU thresholds from 0.5 to 0.95 with an interval of 0.05. We perform ablation studies to analyze different components of our approach, and compare our approach to baselines and state-of-the-art. Pascal VOC07 dataset has 20 object categories. Training is on *trainval* with 5K images and the evaluation is on *test* with 5K images.

5.1. Implementation Details

Our implementation is based on Mask R-CNN benchmark in Pytorch 1.0[1]. Images are resized such that the shortest side is 800 pixels. We use no data augmentation for testing, and only horizontal flipping augmentation for training. The implementation details are described as follows:

[1] https://github.com/facebookresearch/maskrcnn-benchmark

Architecture: Our approach is evaluated on two FPN [26] backbones (ResNet-50 and ResNet-101 [15]), which are pretrained on *ImageNet* [6]. The standard RoI pooling is replaced by RoIAlign [13]. Both heads and RPN are jointly trained end to end. All batch normalization (BN) [19] layers in the backbone are frozen. Each convolution layer in *conv-head* is followed by a BN layer. The bounding box regression is class-specific.

Hyper-parameters: All models are trained using 4 NVIDIA P100 GPUs with 16GB memory, and a mini-batch size of 2 images per GPU. The weight decay is 0.0001 and momentum is 0.9.

Learning Rate Scheduling: All models are fine-tuned with 180k iterations. The learning rate is initialized with 0.01 and reduced to 0.001 after 120K iterations and 0.0001 after 160K iterations.

5.2. Ablation Study

We run a number of ablations to analyze our Double-Head method with ResNet-50 backbone on MS COCO 2017 *val*.

Double-Head Variations: Four variations of double heads are compared:

- **Double-FC** splits the classification and bounding box regression into two fully connected heads, which have the identical structure.
- **Double-Conv** splits the classification and bounding box regression into two convolution heads, which have the identical structure.
- **Double-Head** includes a fully connected head (*fc-head*) for classification and a convolution head (*conv-head*) for bounding box regression.
- **Double-Head-Reverse** switches tasks between two heads (i.e. *fc-head* for bounding box regression and *conv-head* for classification), compared to Double-Head.

The detection performances are shown in Table 1. The top group shows performances for single head detectors. The middle group shows performances for detectors with double heads. The weight for each loss (classification and bounding box regression) is set to 1.0. Compared to the middle group, the bottom group uses different loss weights for *fc-head* and *conv-head* ($\omega^{fc} = 2.0$ and $\omega^{conv} = 2.5$), which are set empirically.

Double-Head outperforms single head detectors by a non-negligible margin (2.0+ AP). It also outperforms Double-FC and Double-Conv by least 1.4 AP. Double-Head-Reverse has the worst performance (drops 6.2+ AP compared to Double-Head). This validates our findings that *fc-head* is more suitable for classification, while *conv-head* is more suitable for localization empirically.

	fc-head		*conv-head*							
	cls	reg	cls	reg	AP	$AP_{0.5}$	$AP_{0.75}$	AP_s	AP_m	AP_l
fc-head	1.0	1.0			36.8	58.7	40.4	21.2	40.1	48.8
conv-head			1.0	1.0	35.9	54.0	39.6	19.1	39.4	50.2
Double-FC	1.0	1.0			37.3	58.7	40.4	21.5	40.1	49.3
Double-Conv			1.0	1.0	33.8	50.5	37.1	16.4	37.2	49.3
Double-Head-Reverse		1.0	1.0		32.6	50.5	35.7	16.2	36.8	46.5
Double-Head	1.0			1.0	**38.8**	**58.9**	**42.3**	**22.1**	**42.2**	**51.3**
Double-FC	2.0	2.0			38.1	59.4	41.3	21.5	41.1	50.0
Double-Conv			2.5	2.5	34.3	51.0	38.0	16.4	38.0	49.0
Double-Head-Reverse		2.0	2.5		32.0	48.8	35.2	14.9	35.2	47.6
Double-Head	2.0			2.5	**39.5**	**59.6**	**43.2**	**22.7**	**42.3**	**52.1**

Table 1. Evaluations of single head and double heads on MS COCO 2017 *val*, using FPN with ResNet-50. The top group shows performances for single head detectors. The middle group shows performances for detectors with double heads. The weight for each loss (classification and bounding box regression) is set to 1.0. Compared to the middle group, the bottom group uses different loss weight for *fc-head* and *conv-head* ($\omega^{fc} = 2.0$, $\omega^{conv} = 2.5$). Clearly, Double-Head has the best performance, outperforming others by a non-negligible margin. Double-Head-Reverse has the worst performance.

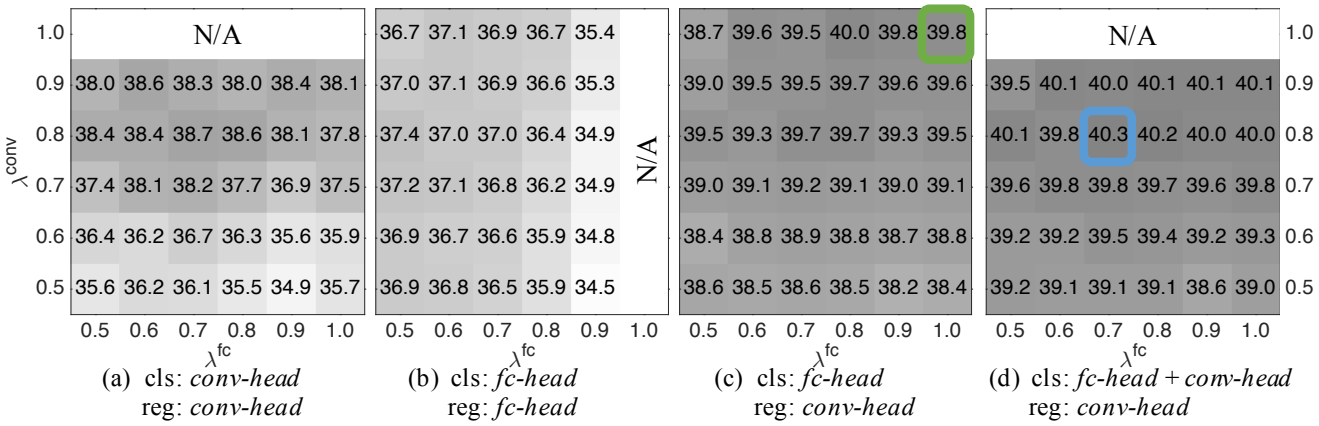

Figure 6. AP over balance weights λ^{fc} and λ^{conv}. For each (λ^{fc}, λ^{conv}) pair, we trained a Double-Head-Ext model. Note that the vanilla Double-Head is a special case with $\lambda^{fc} = 1$, $\lambda^{conv} = 1$. For each model, we evaluate AP in four ways: (a) using *conv-head* alone, (b) using *fc-head* alone, (c) using classification from *fc-head* and bounding box from *conv-head*, and (d) using classification fusion from both heads and bounding box from *conv-head*. Note that the first row in (a) and (d) is not available, due to the unavailability of classification in *conv-head* when $\lambda^{conv} = 1$. The last column in (b) is not available, due to the unavailability of bound box regression in *fc-head* when $\lambda^{fc} = 1$.

Depth of *conv-head*: We study the number of blocks for the two variations of the convolution head (with or without non-local blocks). The evaluations are shown in Table 2. The first group has K residual blocks (Figure 5-(a-b)), while the second group has alternating $(K+1)/2$ residual blocks and $(K-1)/2$ non-local blocks (Figure 5-(c)). We observe that a single block in *conv-head* is slightly behind FPN baseline (drops 0.1 AP) as it is too shallow. However, starting from 2 convolution blocks, the performance boosts substantially (gains 1.9 AP from FPN baseline). As the number of blocks increases, the performance improves gradually with decreasing growth rate. Considering the trade-off between accuracy and complexity, we choose *conv-head* with 3 residual blocks and 2 non-local blocks ($K = 5$ in the second group) for the rest of the paper, which gains 3.0 AP from baseline.

Balance Weights λ^{fc} and λ^{conv}: Figure 6 shows APs for different choices of λ^{fc} and λ^{conv}. For each (λ^{fc}, λ^{conv}) pair, we train a Double-Head-Ext model. The vanilla Double-Head model is corresponding to $\lambda^{fc} = 1$ and $\lambda^{conv} = 1$, while other models involve supervision from unfocused tasks. For each model, we evaluate AP for using *conv-head* alone (Figure 6-(a)), using *fc-head* alone (Figure 6-(b)), using classification from *fc-head* and bounding box from *conv-head* (Figure 6-(c)), and using classification fusion from both heads and bounding box from *conv-head*

NL	K	param	AP	$AP_{0.5}$	$AP_{0.75}$
	0	-	36.8	58.7	40.4
	1	1.06M	36.7 (-0.1)	59.3	39.6
	2	2.13M	38.7 (+1.9)	59.2	41.9
	3	3.19M	39.2 (+2.4)	59.4	42.5
	4	4.25M	39.3 (+2.5)	59.2	42.9
	5	5.31M	39.5 (+2.7)	59.6	43.2
	6	6.38M	39.5 (+2.7)	59.4	43.3
	7	7.44M	39.7 (+2.9)	59.8	43.2
✓	3	4.13M	38.8 (+2.0)	59.2	42.4
✓	5	7.19M	39.8 (+3.0)	59.6	43.6
✓	7	10.25M	**40.0 (+3.2)**	**59.9**	**43.7**

Table 2. The number of blocks (Figure 5) in the convolution head. The baseline ($K = 0$) is equivalent to the original FPN [26] which uses *fc-head* alone. The first group only stacks residual blocks, while the second group alternates $(K + 1)/2$ residual blocks and $(K - 1)/2$ non-local blocks.

Fusion Method	AP	$AP_{0.5}$	$AP_{0.75}$
No fusion	39.7	59.5	43.4
Max	39.9	59.7	43.7
Average	40.1	59.8	44.1
Complementary	**40.3**	**60.3**	**44.2**

Table 3. Fusion of classifiers from both heads. Complementary fusion (Eq. 4) outperforms others. The model is trained using weights $\lambda^{fc} = 0.7$, $\lambda^{conv} = 0.8$.

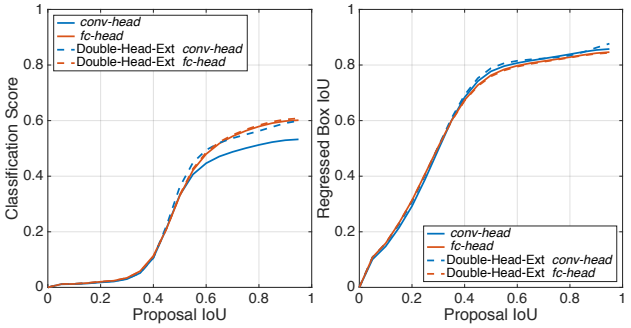

Figure 7. Comparison of classification and bounding box regression between single heads and Double-Head-Ext. Mean of classification scores and regressed box IoUs at different proposal IoUs are at left and right, respectively. The classification scores from *conv-head* in Double-Head-Ext are improved significantly compared to the single *conv-head*. Regression results from *conv-head* of Double-Head-Ext are better as well.

(Figure 6-(d)). ω^{fc} and ω^{conv} are 2.0 and 2.5 in all experiments, respectively.

We summarize key observations as follows: (i) for all models across different (λ^{fc}, λ^{conv}) pairs, using two heads (Figure 6-(c)) outperforms using single head (Figure 6-(a), (b)) by at least 0.9 AP, (ii) for all models, fusion of classifiers introduces at least additional 0.4 AP (compare Figure 6-(c) and (d)), (iii) bounding box regression provides auxiliary in *fc-head* ($\lambda^{fc} = 0.8$ is better than $\lambda^{fc} = 1$ in Figure 6-(c)), and (iv) the best Double-Head-Ext model (with classification fusion) has 40.3 AP, corresponding to $\lambda^{fc} = 0.7$, $\lambda^{conv} = 0.8$ (blue box in Figure 6-(d)). It outperforms Double-Head (39.8 AP, green box in Figure 6-(c)) by 0.5 AP. For the rest of the paper, we use $\lambda^{fc} = 0.7$ and $\lambda^{conv} = 0.8$ for Double-Head-Ext.

Fusion of Classifiers: We study three different ways to fuse the classification scores from both the fully connected head (s^{fc}) and the convolution head (s^{conv}) during inference: (a) average, (b) maximum, and (c) complementary fusion using Eq. (4). The evaluations are shown in Table 3. The proposed complementary fusion outperforms other fusion methods (max and average) and gains 0.6 AP compared to using the score from *fc-head* alone.

5.3. Comparison with Baselines

We compare our method with Faster RCNN [34] and FPN [26] baselines on two backbones (ResNet-50 and ResNet-101). Note that FPN baseline only has *fc-head*. Our method has two variations (Double-Head and Double-Head-Ext).

The evaluations on MS COCO 2017 *val* are shown in Table 4. Our method outperforms both FPN and Faster RCNN baselines on *all* evaluation metrics. Compared with FPN baselines, our Double-Head-Ext gains 3.5 and 2.8 of AP on ResNet-50 and ResNet-101 backbones, respectively. Our method gains 3.5+ AP on the higher IoU threshold (0.75) and 1.4+ AP on the lower IoU threshold (0.5) for both backbones. This demonstrates the advantage of our method with double heads.

We also observe that Faster R-CNN and FPN have different preferences over object sizes when using ResNet-101 backbone: i.e. Faster R-CNN has better AP on medium and large objects, while FPN is better on small objects. Even comparing with the best performance among FPN and Faster R-CNN across different sizes, our Double-Head-Ext gains 1.7 AP on small objects, 2.1 AP on medium objects and 2.5 AP on large objects. This demonstrates the superiority of our method, which leverages the advantage of *fc-head* on classification and the advantage of *conv-head* on localization. Qualitative results are in supplementary materials.

To examine the effect of jointly training with both heads, we conduct the sliding windows analysis (details in section 3.1) on Double-Head-Ext with the ResNet-50 backbone and compare the results on all objects with two single head models. Results in Figure 7 show: (i) the classification scores from *conv-head* in Double-Head-Ext are improved significantly compared with using *conv-head* alone and (ii) regression results from *conv-head* of Double-Head-Ext are better

Method	Backbone	AP	$AP_{0.5}$	$AP_{0.75}$	AP_s	AP_m	AP_l
Faster R-CNN [34]	ResNet-50-C4	34.8	55.8	37.0	19.1	38.8	48.2
FPN baseline [26]	ResNet-50	36.8	58.7	40.4	21.2	40.1	48.8
Double-Head (ours)	ResNet-50	39.8	59.6	43.6	**22.7**	42.9	53.1
Double-Head-Ext (ours)	ResNet-50	**40.3**	**60.3**	**44.2**	22.4	**43.3**	**54.3**
Faster R-CNN [34]	ResNet-101-C4	38.5	59.4	41.4	19.7	43.1	53.3
FPN baseline [26]	ResNet-101	39.1	61.0	42.4	22.2	42.5	51.0
Double-Head (ours)	ResNet-101	41.5	61.7	45.6	23.8	**45.2**	54.9
Double-Head-Ext (ours)	ResNet-101	**41.9**	**62.4**	**45.9**	**23.9**	**45.2**	**55.8**

Table 4. Comparisons with baselines (FPN [26] and Faster RCNN [34]) on MS COCO 2017 *val*. Note that FPN baseline only has *fc-head*. Our Double-Head and Double-Head-Ext outperform both Faster R-CNN and FPN baselines on two backbones (ResNet-50 and ResNet-101).

Method	Backbone	AP	$AP_{0.5}$	$AP_{0.75}$	AP_s	AP_m	AP_l
FPN [26]	ResNet-101	36.2	59.1	39.0	18.2	39.0	48.2
Mask RCNN [13]	ResNet-101	38.2	60.3	41.7	20.1	41.1	50.2
Deep Regionlets [43]	ResNet-101	39.3	59.8	-	21.7	43.7	50.9
IOU-Net [20]	ResNet-101	40.6	59.0	-	-	-	-
Soft-NMS [2]	Aligned-Inception-ResNet	40.9	**62.8**	-	23.3	43.6	53.3
LTR [38]	ResNet-101	41.0	60.8	44.5	23.2	44.5	52.5
Fitness NMS [40]	DeNet-101 [39]	41.8	60.9	44.9	21.5	**45.0**	**57.5**
Double-Head-Ext (ours)	ResNet-101	**42.3**	**62.8**	**46.3**	**23.9**	44.9	54.3

Table 5. Comparisons with state-of-the-art single-model detectors on MS COCO 2017 *test-dev*. Our Double-Head-Ext achieves the best performance among these two-stage detectors with one training stage.

Method	AP	$AP_{0.5}$	$AP_{0.75}$
FPN baseline [26]	47.4	75.7	41.9
Double-Head-Ext (ours)	**49.2**	**76.7**	**45.6**

Table 6. Comparisons with FPN baseline [26] on VOC 07 datasets with ResNet-50 backbone. Our Double-Head-Ext outperforms FPN baseline.

as well. We believe that these improvements come from the jointly training of two heads with the shared backbone.

We also conduct experiments on Pascal VOC07 dataset and results are shown in Table 6. Our method gains 1.8 in AP, 1.0 in $AP_{0.5}$ and 3.7 in $AP_{0.75}$ compared with FPN baseline.

5.4. Comparison with State-of-the-art

We compare our Double-Head-Ext using ResNet-101 backbone with state-of-the-art on MS COCO 2017 *test-dev* in Table 5. We report performance of all methods with single-model inference for fair comparison. Our Double-Head-Ext achieves the best performance with 42.3 AP compared with state-of-the-art two-stage detectors with one training stage. Cascade R-CNN [3] is a single-model inference method but involves multiple training stages. Compared with Cascade R-CNN using ResNet-101 backbone with 42.8 AP, our Double-Head-Ext is comparable even that our method only has one training stage. These results demonstrate the superior performance of the proposed method.

6. Conclusions

In this paper, we performed a thorough analysis and found an interesting fact that two widely used head structures (convolution head and fully connected head) have opposite preferences towards classification and localization tasks in object detection. Specifically, *fc-head* is more suitable for the classification task, while *conv-head* is more suitable for the localization task. Furthermore, we examined the weight matrix in the *fc-head* and found that it learns spatial sensitive transformations. We believe that this allows *fc-head* to tell apart a complete object from part of an object, but is not robust to regress the whole object. Based upon these findings, we proposed a Double-Head method, which has a fully connected head focusing on classification and a convolution head for bounding box regression. Without bells and whistles, our method gains +3.5 and +2.8 AP on MS COCO dataset from FPN baselines with ResNet-50 and ResNet-101 backbones, respectively, and achieves state-of-the-art. We believe that our rethinking, findings and results are helpful for future research in object detection.

References

[1] Mscoco instance segmentation challenges 2018 megvii (face++) team. http://presentations.cocodataset.org/ECCV18/COCO18-Detect-Megvii.pdf, 2018. 2

[2] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*, pages 5561–5569, 2017. 8

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2, 5, 8

[4] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems 29*, pages 379–387. 2016. 1, 2

[5] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 764–773, 2017. 2

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[7] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2

[8] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 5

[9] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017. 2

[10] Ross Girshick. Fast R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 1, 2

[11] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1

[12] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 2

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017. 2, 4, 5, 8

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European conference on computer vision*, pages 346–361. Springer, 2014. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2, 4, 5

[16] Yihui He, Chenchen Zhu, Jianren Wang, Marios Savvides, and Xiangyu Zhang. Bounding box regression with uncertainty for accurate object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2888–2897, 2019. 2

[17] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 2

[18] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask Scoring R-CNN. In *CVPR*, 2019. 2

[19] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015. 5

[20] Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 784–799, 2018. 2, 8

[21] Hei Law and Jia Deng. CornerNet: Detecting Objects as Paired Keypoints. In *ECCV*, 2018. 2

[22] Hei Law, Yun Teng, Olga Russakovsky, and Jia Deng. Cornernet-lite: Efficient keypoint based object detection. *arXiv preprint arXiv:1904.08900*, 2019. 2

[23] Ang Li, Xue Yang, and Chongyang Zhang. Rethinking classification and localization for cascade r-cnn. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2019. 2

[24] Yanghao Li, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Scale-aware trident networks for object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2

[25] Zeming Li, Chao Peng, Gang Yu, Xiangyu Zhang, Yangdong Deng, and Jian Sun. Light-head r-cnn: In defense of two-stage object detector. *arXiv preprint arXiv:1711.07264*, 2017. 2

[26] T. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944, July 2017. 1, 2, 4, 5, 7, 8

[27] Tsung-Yi Lin, Priyal Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE transactions on pattern analysis and machine intelligence*, 2018. 2

[28] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 2, 5

[29] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C

Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 2

[30] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 116–131, 2018. 2

[31] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 2

[32] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. 2

[33] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 2

[34] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 1, 2, 3, 7, 8

[35] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 2

[36] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Robert Fergus, and Yann Lecun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, 2014. 2

[37] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 2

[38] Zhiyu Tan, Xuecheng Nie, Qi Qian, Nan Li, and Hao Li. Learning to rank proposals for object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2, 8

[39] Lachlan Tychsen-Smith and Lars Petersson. Denet: Scalable real-time object detection with directed sparse sampling. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 428–436, 2017. 8

[40] Lachlan Tychsen-Smith and Lars Petersson. Improving object localization with fitness nms and bounded iou loss. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6877–6885, 2018. 8

[41] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013. 2

[42] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018. 4

[43] Hongyu Xu, Xutao Lv, Xiaoyu Wang, Zhou Ren, Navaneeth Bodla, and Rama Chellappa. Deep regionlets for object detection. In *The European Conference on Computer Vision (ECCV)*, September 2018. 8

[44] Haichao Zhang and Jianyu Wang. Towards adversarially robust object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2

[45] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6848–6856, 2018. 2

[46] Xingyi Zhou, Dequan Wang, and Philipp Krahenbuhl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 2

[47] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krähenbühl. Bottom-up object detection by grouping extreme and center points. In *CVPR*, 2019. 2

[48] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. *CVPR*, 2019. 2

APPENDIX

In this appendix, we firstly visualize the spatial correlation over feature maps for both the fully connected head (*fc-head*) and the convolution head (*conv-head*). Then, we provide qualitative analysis to compare *fc-head* with *conv-head*.

A. Spatial Correlation of Feature Maps in Heads

We demonstrate that *conv-head* and *fc-head* have different spatial sensitivity in Figure 8. We analyze the spatial correlation of input and output feature maps for both heads. The spatial correlation of a feature map is calculated in the similar manner to the spatial correlation of the weight matrix in the first fully connected layer of *fc-head*, which is discussed in section 3.4. For a given feature map (a 7×7 grid), we first compute correlation between each pair of locations using the cosine distance between two feature vectors. This results in a 49×49 correlation matrix (one row per cell). Then we convert each row into a 7×7 matrix to show the correlation between a cell and all other cells by keeping their 2D relationship.

The first column of Figure 8 shows *strong* spatial correlation of input feature maps for both heads. We also compute the spatial correlation for output feature maps (shown in the second column of Figure 8). The output feature map is not available for *fc-head* as the first fully connected layer outputs a feature vector with dimension 1024. We reconstruct the output feature maps for the *fc-head* by splitting the weight matrix of fully connected layer ($256 \cdot 7 \cdot 7 \times 1024$) into 49 matrices with dimension 256×1024. Each matrix is used for a cell in the input feature map ($7 \times 7 \times 256$). Thus we reconstruct the output feature map $7 \times 7 \times 1024$ for *fc-head*. Output feature map of *fc-head* has significant less spatial correlation than *conv-head*. We believe this is due to the spatial sensitivity of the weight matrix for the first fully connected layer in *fc-head*, which is discussed in the submission draft section 3.4.

B. Qualitative Analysis

We apply a well trained Double-Head-Ext model (Figure 1-(d) in the submission draft) and compare the detection results of (a) using *conv-head* alone, (b) using *fc-head* alone, and (c) using both heads.

Figure 9 shows three cases that *fc-head* is better than *conv-head* in classification. In all three cases, *conv-head* misses small objects due to the low classification scores (e.g. signal light in case I, cows in case II, and persons in case III). In contrast, these objects are successfully detected by using *fc-head* (in the green box) with proper classification scores. Our Double-Head-Ext successfully detects these objects, by leveraging the superiority of *fc-head* for classification.

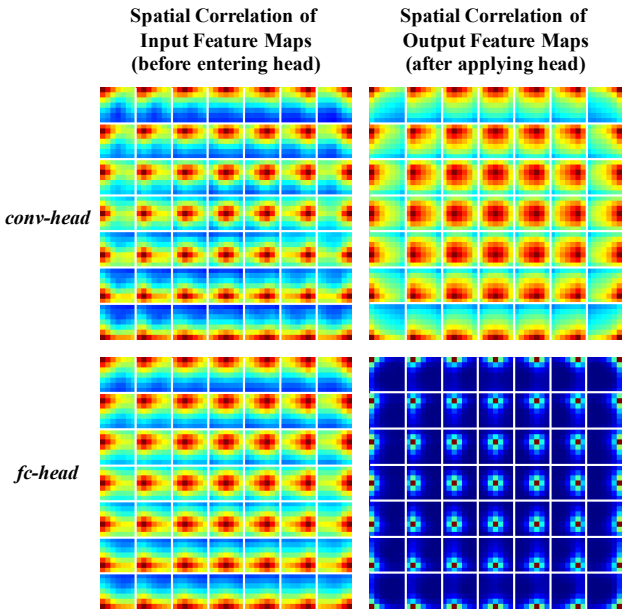

Figure 8. Spatial correlation of feature maps in both *fc-head* and *conv-head*. Top-row: spatial correlation of input and output feature maps in *conv-head*. Bottom row: spatial correlation of input and output feature maps in *fc-head*. The input feature maps have strong spatial correlation for both *conv-head* and *fc-head*. However *fc-head* has significant less spatial correlation on the output feature map than *conv-head*. We believe this is due to the spatial sensitivity of the weight matrix for the first fully connected layer in *fc-head*.

Figure 10 demonstrates two cases that *conv-head* is better than *fc-head* in localization. Compared with *fc-head* which has a duplicate detection for both cases (i.e. baseball bat in case I, and surfing board in case II, shown in the red box at the bottom row), *conv-head* has a single accurate detection (in the green box at the bottom row). Both heads share proposals (yellow boxes). The duplicated box from *fc-head* comes from an inaccurate proposal. It is not suppressed by NMS as it has low IoU with other boxes around the object. In contrast, *conv-head* has a more accurate regression box for the same proposal, which has higher IoU with other boxes. This allows NMS to remove it. Double-Head-Ext has no duplication, by leveraging the superiority of *conv-head* in localization.

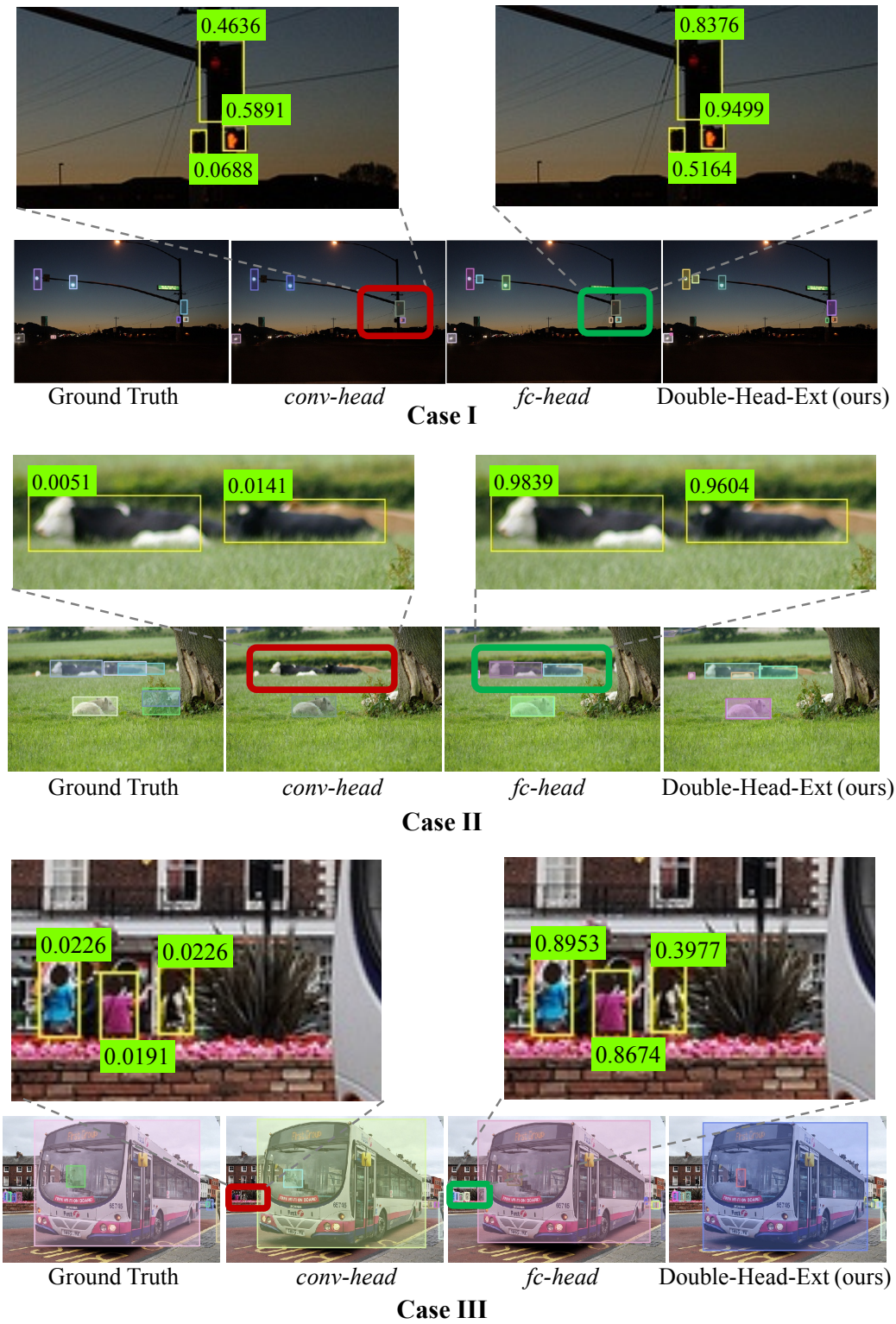

Figure 9. *fc-head* head is more suitable for classification than *conv-head*. This figure includes three cases. Each case has two rows. The bottom row shows ground truth, detection results using *conv-head* alone, *fc-head* alone, and our Double-Head-Ext (from left to right). The *conv-head* misses objects in the red box. In contrast, these missing objects are successfully detected by *fc-head* (shown in the corresponding green box). The top row zooms in the red and green boxes, and shows classification scores from the two heads for each object. The missed objects in *conv-head* have small classification scores, compared to *fc-head*.

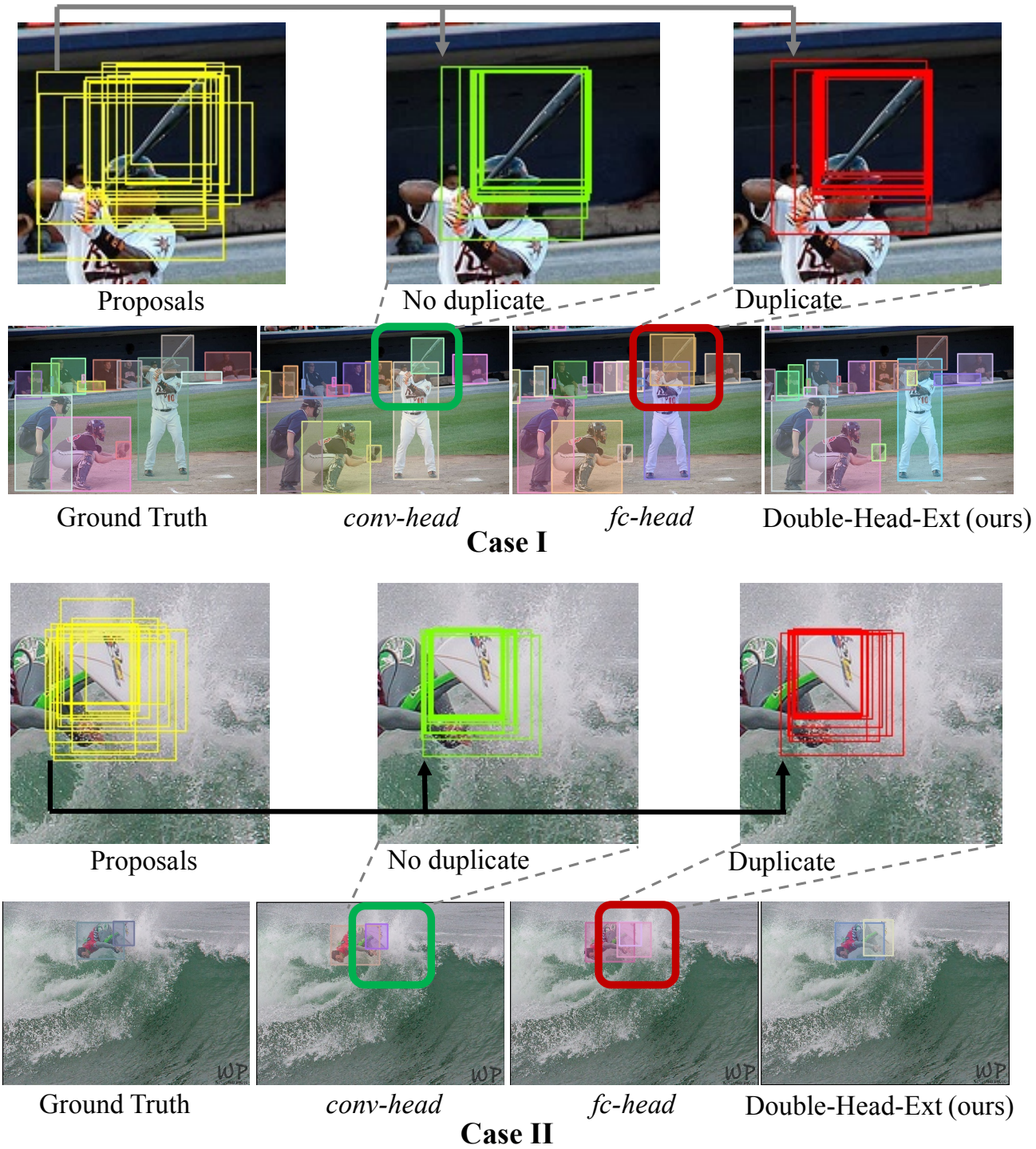

Figure 10. *conv-head* is more suitable for localization than *fc-head*. This figure includes two cases. Each case has two rows. The bottom row shows ground truth, detection results using *conv-head* alone, *fc-head* alone, and our Double-Head-Ext (from left to right). *fc-head* has a duplicate detection for the baseball bat in case I and for the surfing board in case II (in the red box). The duplicate detection is generated from an inaccurate proposal (shown in the top row), but is not removed by NMS due to its low IoU with other detection boxes. In contrast, *conv-head* has more accurate box regression for the same proposal, with higher IoU with other detection boxes. Thus it is removed by NMS, resulting no duplication.