结论：fc head更适合分类任务，conv head更适合回归任务

本文使用两个head做两个任务， conv head使用的是resnet block以及non local block, fc head使用2个fc层

Double-Head RCNN: 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

Modern R-CNN based detectors apply a head to extract Region of Interest (RoI) features for both classification and localization tasks. In contrast, we found that these two tasks have **opposite** preferences towards two widely used head structures (i.e. fully connected head and convolution head). Specifically, the fully connected head is more suitable for the classification task, while the convolution head is more suitable for the localization task. Therefore, we propose a **Double-Head** method, which has a fully connected head focusing on classification and a convolution head to pay more attention to bounding box regression.

In addition, we have two findings for the unfocused tasks (i.e. classification in the convolution head, and bounding box regression in the fully connected head): (a) adding classification to the convolution head is complementary to the classification in the fully connected head, and (b) bounding box regression provides auxiliary supervision for the fully connected head. 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

A majority of two-stage object detectors [8; 9; 28; 3; 21] share the head network for two tasks (classification and bounding box regression). Two different head structures are widely used: a convolution head (conv5) on single feature map (conv4) in Faster R-CNN [28] and a fully connected head (2-*fc*) on multiple level feature maps in FPN [21]. However, there is a lack of understanding of the correlation between the two tasks and the two head structures.

Related to this problem, recent COCO Detection 18 Challenge winner (Megvii[1]) found that, for instance segmentation, it is better to couple bounding box regression with segmentation on a convolution head than with classification on a fully connected head. This motivates us to rethink the classification and bounding box regression for object detection, with respect to different head structures. Intuitively, spatial information is crucial for object classification to determine if a complete object (not just part of the object) is covered by the region proposal. The fully connected head fits well for this task, as it is spatial sensitive. In contrast, the bounding box regression task requires the object level context to determine the bounding box offset. The convolution is more suitable for this task due to its capability to extract object level context. Therefore, we believe that neither a single fully connected head nor a single convolution head is good enough to handle classification and localization simultaneously.

In this paper, 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 the advantage of both heads. This design outperforms both single *fc-head* and single

[1]`http://cocodataset.org/#detection-leaderboard`

Preprint. Under review.

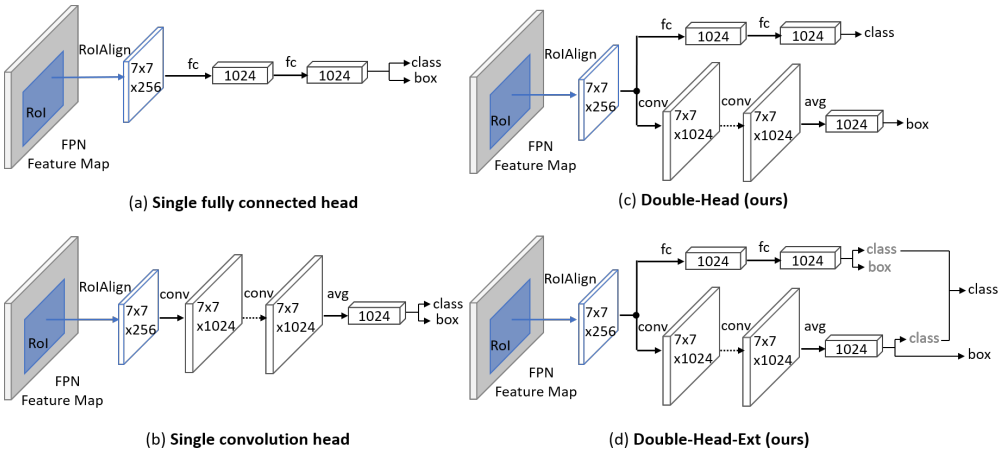


Figure 1: Comparison between single head and double heads. (a) single fully connected (2-*fc*) head, (b) single convolution head, (c) Double-Head method, which splits classification and localization on a fully connected head and a convolution head respectively, (d) Double-Head-Ext, which extends Double-Head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

conv-head (see Figure 1-(a), (b)) by a non-negligible margin. This is based upon our finding that *fc-head* prefers the classification task while *conv-head* prefers the localization task.

In addition, we propose Double-Head-Ext (Figure 1-(d)), an extension of Double-Head, to further improve the accuracy by leveraging unfocused tasks (i.e. classification in *conv-head*, and bounding box regression in *fc-head*). This is based upon two findings on the unfocused tasks: (a) adding classification to *conv-head* is complementary to the classification in *fc-head*, and (b) bounding box regression provides auxiliary supervision for *fc-head*.

Both Double-Head and Double-Head-Ext outperform FPN baseline by a non-negligible margin. Experimental results on MS COCO dataset demonstrates that our best model (Double-Head-Ext) gains 3.5 and 2.8 AP over FPN baselines with ResNet-50 and ResNet-101 backbones, respectively.

2 Related Work

One-stage Object Detectors: OverFeat [30] detects objects by sliding windows on feature maps. SSD [24; 7] and YOLO [27; 26] have been tuned for speed by predicting object classes and locations directly. RetinaNet [23] alleviates the extreme foreground-background class imbalance problem by introducing focal loss. [17; 18; 35; 6; 36] model an object as a single point or a set of keypoints (corner, center, etc), and build on the robust keypoint estimation network.

Two-stage Object Detectors: RCNN [10] applies a deep neural network to extract features for proposals generated by selective search [32]. SPPNet [11] speeds up RCNN significantly by using spatial pyramid pooling. Fast RCNN [8] improves the speed and performance by utilizing a differentiable RoI Pooling. Faster RCNN [28] introduces Region Proposal Network (RPN) into the network. R-FCN [3] employs position sensitive RoI pooling to address the translation-variance problem. FPN [21] builds a top-down architecture with lateral connections to detect proposals across multiple layers.

Backbone Networks: Fast RCNN [8] and Faster RCNN [28] extract features from stage conv4 of VGG-16 [31], while FPN [21] utilizes features from multiple layers (conv2 to conv5) of ResNet [12]. Deformable ConvNet [4; 37] proposes deformable convolution and deformable RoI pooling to augment spatial sampling locations. Trident Network [19] generates scale-aware feature maps with multi-branch architecture. MobileNet [14; 29] and ShuffleNet [34; 25] introduced efficient operators (like depth-wise convolution, group convolution, channel shuffle, etc) to speed up on mobile devices.

Detection Heads: Light-Head RCNN [20] introduces an efficient head network with thin feature maps. Cascade RCNN [2] constructs a sequence of detector heads trained with increasing IoU threshold. Mask RCNN [13] introduces an extra head for instance segmentation. IoU-Net [16]

introduces a branch to predict IoU between detected bounding box and ground-truth. Similar to IoU Net, Mask Scoring RCNN [15] presents an extra head to predict MaskIoU score for each segmentation mask. In contrast to the existing methods, which apply a head to extract RoI features for both classification and bounding box regression, we propose to split these two tasks into different heads.

3 Our Approach: Double-Head

In this section, we discuss our Double-Head method in details. Firstly, we introduce our motivation and hypothesis. Then 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. Furthermore, 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*).

3.1 Motivation and Hypothesis

Motivation: COCO Detection 18 Challenge winner (Megvii) showed that, for instance segmentation, it is better to decouple bounding box regression from classification in the fully connected head and couple it with segmentation in the convolution head. This motivates us to rethink the classification and bounding box regression tasks for object detection with respect to different head structures.

Intuitively, we believe that the convolution head is more suitable for bounding box regression, even *without* the help from instance segmentation. This is because the convolution head is able to capture the context for the whole object, which is crucial to refine the bounding box from the proposal. In contrast, the fully connected head, which is spatial sensitive, fits well for the object classification. Different from image classification, object classification needs to consider if the object (if there is) is *complete* within the proposed bounding box. This requires both context and spatial information.

Hypothesis: Given the motivation discussed above, we have a hypothesis as follows:

> *The fully connected head is more suitable for object classification, while the convolution head is more suitable for object localization.*

Validation: To validate this hypothesis, we propose a new Double-Head design that includes a fully connected head (*fc-head*) for classification and a convolution head (*conv-head*) for bounding box regression (see Figure 1-(c)), on a shared backbone network. To complete the comparison, we introduce Double-Head-Reverse, which switches tasks between two heads (i.e. *fc-head* for bounding box regression and *conv-head* for classification). Here, we use FPN on ResNet-50 as backbone and stack five residual blocks in *conv-head* (details will be shown later in Sec 3.2).

The detection performances on MS COCO 2014 *minival* are shown in Table 5. Double-Head outperforms detectors with single head by a non-negligible margin (2.7+ AP). When reversing tasks between the two heads (Double-Head-Reverse), AP drops 7.5 points. This validates our hypothesis that *fc-head* is more suitable for classification, while *conv-head* is more suitable for localization. Next, we will discuss Double-Head design.

3.2 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 [21] backbone to generate region proposals and extract object features from multiple levels using RoI align [13]. Each proposal has a feature map with size $7 \times 7 \times 256$, which is transformed by *fc-head* and *conv-head* into two feature vectors (each with 1024 dimensions) 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 [21] (Figure 1-(a)). The output dimension is 1024. The parameter size is 13.25M.

Convolution Head (*conv-head*) stacks K residual blocks [12]. The first block increases the number of channels from 256 to 1024 (shown in Figure 2-(a)), and others are bottleneck blocks (shown in Figure 2-(b)). At the end, average pooling is used to generate the feature vector with dimension

	fc-head		*conv-head*							
	cls	reg	cls	reg	AP	$AP_{0.5}$	$AP_{0.75}$	AP_s	AP_m	AP_l
Single-Conv			✓	✓	35.9	54.0	39.6	19.1	39.4	50.2
Single-FC	✓	✓			36.8	58.7	40.4	21.2	40.1	48.8
Double-Head-Reverse		✓	✓		32.0	48.8	35.2	14.9	35.2	47.6
Double-Head	✓			✓	**39.5**	**59.6**	**43.2**	**22.7**	**42.3**	**52.1**

Table 1: Evaluations of single head and double heads on MS COCO 2014 *minival*, using FPN on ResNet-50. Double-Head-Reverse switches tasks between the two heads, compared to Double-Head.

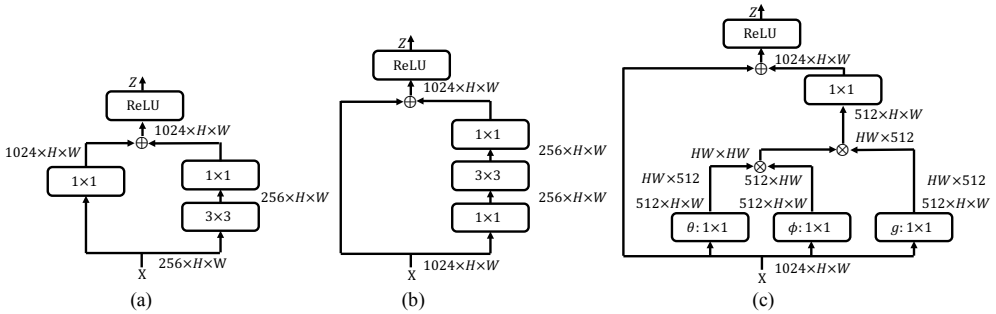


Figure 2: Network architectures of three components: (a) residual block to increase the number of channels (from 256 to 1024), (b) residual bottleneck block, and (c) non-local block.

1024. Each residual block has 1.06M parameters. We also introduce a variation for the convolution head by inserting a non-local block [33] (see Figure 2-(c)) before each bottleneck block to enhance foreground. 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.

3.3 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) the 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:

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

where L^{conv}_{cls} and L^{conv}_{reg} are the 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^{conv}_{reg},

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 [21], 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:

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

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 [2] which combining all classifiers by average. Note that this fusion is only applicable when $\lambda^{fc} \neq 0$ and $\lambda^{conv} \neq 1$.

4 Experimental Results

We evaluate our approach on MS COCO 2014 dataset [22], which has 80 object categories. Following [1; 21], the union of *train* with 80K images and *val35k* with 35K images is used as the training set. The evaluation is on *minival* with 5K 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 the original FPN [21] with single 2-*fc* head (referred as FPN baseline).

4.1 Implementation Details

Our implementation is based on Mask R-CNN benchmark in Pytorch 1.0[2]. 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:

Architecture: Our approach is evaluated on two FPN [21] backbones (ResNet-50 and ResNet-101 [12]), which are pretrained on *ImageNet* [5]. The standard RoI pooling is replaced by RoIAlign [13]. Both heads and RPN are jointly trained. All BN 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 in 4 NVIDIA P100 GPUs with 16GB memory, with a mini-batch size of 2 images per GPU. The weight decay is 0.0001 and momentum is 0.9. The weights for the two heads are empirically set as $\omega^{fc} = 2.0$ and $\omega^{conv} = 2.5$.

Learning rate scheduling: All models were 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.

4.2 Ablation Study

We run a number of ablations to analyze our Double-Head method with ResNet-50 backbone.

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 2-(a-b)), while the second group has alternating $(K + 1)/2$ residual blocks and $(K - 1)/2$ non-local blocks (Figure 2-(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 3 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 3-(a)), using *fc-head* alone

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

NL	K	param	AP	$AP_{0.5}$	$AP_{0.75}$	AP_s	AP_m	AP_l
	0 (baseline)	-	36.8	58.7	40.4	21.2	40.1	48.8
	1	1.06M	36.7 (-0.1)	59.3	39.6	21.1	39.5	48.1
	2	2.13M	38.7 (+1.9)	59.2	41.9	21.5	41.5	51.6
	3	3.19M	39.2 (+2.4)	59.4	42.5	22.2	42.2	51.6
	4	4.25M	39.3 (+2.5)	59.2	42.9	22.0	42.2	52.6
	5	5.31M	39.5 (+2.7)	59.6	43.2	22.7	42.3	52.1
	6	6.38M	39.5 (+2.7)	59.4	43.3	**22.8**	42.3	52.4
	7	7.44M	39.7 (+2.9)	59.8	43.2	22.2	42.6	52.7
✓	3	4.13M	38.8 (+2.0)	59.2	42.4	21.9	41.7	51.6
✓	5	7.19M	39.8 (+3.0)	59.6	43.6	22.7	**42.9**	53.1
✓	7	10.25M	**40.0 (+3.2)**	**59.9**	**43.7**	22.3	42.8	**53.4**

Table 2: The number of blocks (Figure 2) in the convolution head. The baseline ($K = 0$) is equivalent to the original FPN [21] 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.

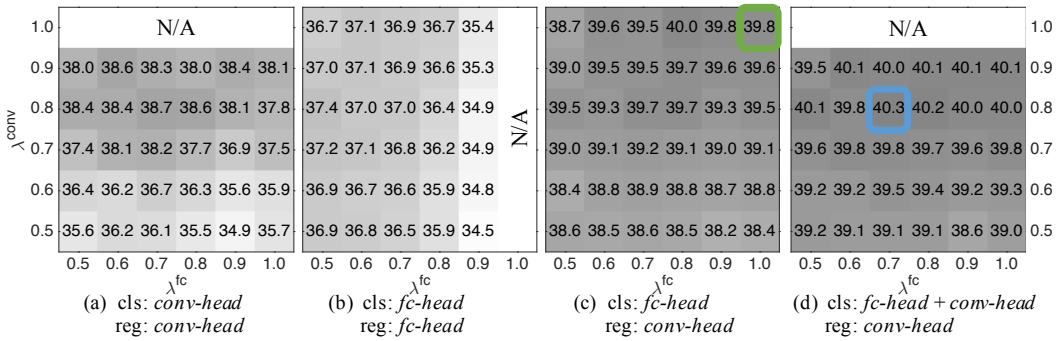


Figure 3: AP over balance weights λ^{fc} and λ^{conv}. For each (λ^{fc}, λ^{conv}) pair, we trained a Double-Head-Ext model. Note that the vanilla Double-Head is a special case with $\lambda^{fc} = 1$, $\lambda^{conv} = 1$. For each model, we evaluate AP in four ways: (a) using *conv-head* alone, (b) 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 $\lambda^{fc} = 1$.

Fusion Method	Score	AP	$AP_{0.5}$	$AP_{0.75}$	AP_s	AP_m	AP_l
No fusion	s^{fc}	39.7	59.5	43.4	21.9	42.8	52.9
Max	$max(s^{fc}, s^{conv})$	39.9	59.7	43.7	22.1	43.0	53.5
Average	$(s^{fc} + s^{conv})/2$	40.1	59.8	44.1	22.2	43.0	54.0
Complementary	$s^{fc} + s^{conv}(1 - s^{fc})$	**40.3**	**60.3**	**44.2**	**22.4**	**43.3**	**54.3**

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 3-(b)), using classification from *fc-head* and bounding box from *conv-head* (Figure 3-(c)), and using classification fusion from both heads and bounding box from *conv-head* (Figure 3-(d)).

We summarize key observations as follows: (i) for all models across different (λ^{fc}, λ^{conv}) pairs, using two heads (Figure 3-(c)) outperforms using single head (Figure 3-(a), (b)) by at least 0.9 AP, (ii) for all models, fusion of classifiers introduces at least additional 0.4 AP (compare Figure 3-(c) and (d)), (iii) bounding box regression provides auxiliary in *fc-head* ($\lambda^{fc} = 0.8$ is better than $\lambda^{fc} = 1$ in Figure 3-(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 3-(d)). It outperforms Double-Head (39.8 AP, green box in Figure 3-(c)) by 0.5 AP. For the rest of the paper, we use $\lambda^{fc} = 0.7$, $\lambda^{conv} = 0.8$ for Double-Head-Ext.

Method	Backbone	AP	$AP_{0.5}$	$AP_{0.75}$	AP_s	AP_m	AP_l
Faster R-CNN [28]	ResNet-50-C4	34.8	55.8	37.0	19.1	38.8	48.2
FPN baseline [21]	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 [28]	ResNet-101-C4	38.5	59.4	41.4	19.7	43.1	53.3
FPN baseline [21]	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 [21] and Faster RCNN [28]) on MS COCO 2014 *minival*. 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).

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.

4.3 Comparison with Faster R-CNN and FPN Baselines

We compare our Double-Head with Faster RCNN [28] and FPN [21] 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). Both variations has three residual blocks and two non-local blocks in *conv-head*. The balance weights for Double-Head are set as $\lambda^{fc} = 1$ and $\lambda^{conv} = 1$. For Double-Head-Ext, the balance weights are set as $\lambda^{fc} = 0.7$ and $\lambda^{conv} = 0.8$.

The evaluations on MS COCO 2014 *minival* are shown in Table 4. Our method outperforms both FPN and Faster RCNN baselines on *all* evaluation metrics. Compared to 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.

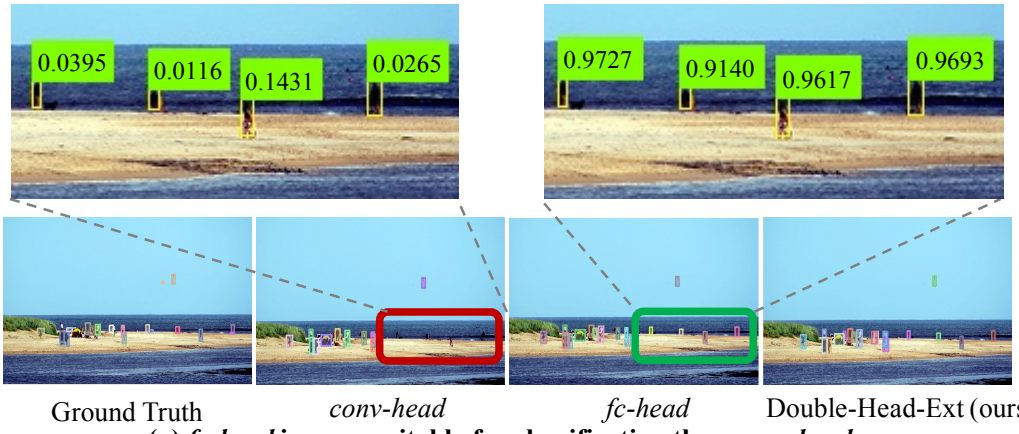
We also observe that Faster R-CNN and FPN have different preferences over object scales when using ResNet-101 backbone: i.e. Faster R-CNN has better AP on medium and large objects, while FPN is better on small objects. Even comparing with the best performance among FPN and Faster R-CNN across different scales, 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.

4.4 Qualitative Analysis

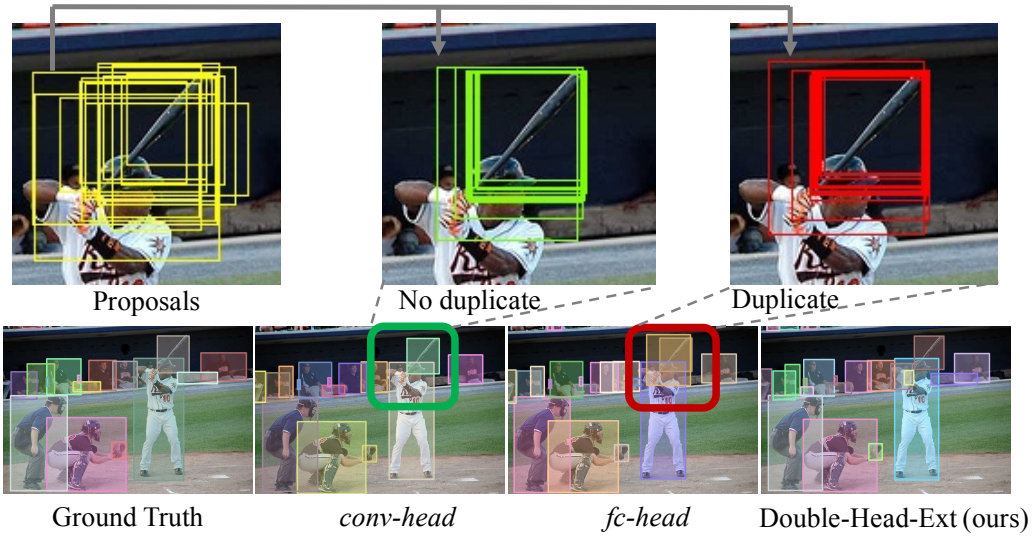
In this section, we compare *fc-head* with *conv-head* qualitatively. We apply a well trained Double-Head-Ext model (Figure 1-(d)) and compare the detection results of (a) using *conv-head* alone, (b) using *fc-head* alone, and (c) using both heads.

Figure 4-(a) illustrates the superiority of *fc-head* on classification over *conv-head*. Four small objects are missed when using *conv-head* alone (in the red box at the bottom row) due to their small classification scores shown in the top row (zoom-in view). 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 4-(b) demonstrates the advantage of *conv-head* in localization. Compared with *fc-head* which has a duplicate detection for the baseball bat (in the red box at the bottom row), *conv-head* has a single detection (in the green box at the bottom row). Both heads share proposals (yellow boxes). The duplicated box from *fc-head* comes from the biggest proposal. It is not suppressed by NMS as it has low IoU with other boxes around the baseball bat. In contrast, *conv-head* has more accurate regression result for the same proposal, which has higher IoU with other boxes. This allows NMS to remove it. Double-Head-Ext has no duplication, leveraging the superiority of *conv-head* in localization.



(a) *fc-head* is more suitable for classification than *conv-head*.



(b) *conv-head* is more suitable for localization than *fc-head*.

Figure 4: Qualitative comparison between *conv-head* and *fc-head*. **(a) *fc-head* is more suitable for classification than *conv-head***: the ground truth and detection results of using *conv-head* alone, *fc-head* alone, and our Double-Head-Ext are shown in the bottom row. *conv-head* misses several small objects (in the red box) due to their small classification scores (shown in the top row). These objects are successfully detected by *fc-head* with proper scores. **(b) *conv-head* is more suitable for localization than *fc-head***: similarly, the ground truth and detection results are shown in the bottom row. *fc-head* has a duplicate detection for the baseball bat (in the red box). It 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.

5 Conclusions

In this paper, we validated our hypothesis that the fully connected head (*fc-head*) is more suitable for object classification, while the convolution head (*conv-head*) is more suitable for object localization. Based on this finding, we proposed to separate classification and bounding box regression into *fc-head* and *conv-head*, respectively. Furthermore, we found that unfocused tasks can help in two aspects: (a) object classification is enhanced by adding classification task in *conv-head*, as it is complementary to the classification in *fc-head*, and (b) bounding box regression provides auxiliary supervision for *fc-head*. Our method gains 3.5 and 2.8 AP over FPN with ResNet-50 and ResNet-101 respectively on MS COCO. We believe this is helpful for future research in object detection.

References

[1] S. Bell, C. Lawrence Zitnick, K. Bala, and R. Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2874–2883, 2016.

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

[3] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems 29*, pages 379–387. 2016. URL `http://papers.nips.cc/paper/6465-r-fcn-object-detection-via-region-based-fully-convolutional-networks.pdf`.

[4] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 764–773, 2017.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[6] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian. Centernet: Keypoint triplets for object detection. *arXiv preprint arXiv:1904.08189*, 2019.

[7] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017.

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

[9] R. Girshick, J. Donahue, T. Darrell, and J. 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.

[10] R. Girshick, J. Donahue, T. Darrell, and J. 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.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European conference on computer vision*, pages 346–361. Springer, 2014.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[14] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[15] Z. Huang, L. Huang, Y. Gong, C. Huang, and X. Wang. Mask Scoring R-CNN. In *CVPR*, 2019.

[16] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 784–799, 2018.

[17] H. Law and J. Deng. CornerNet: Detecting Objects as Paired Keypoints. In *ECCV*, 2018.

[18] H. Law, Y. Teng, O. Russakovsky, and J. Deng. Cornernet-lite: Efficient keypoint based object detection. *arXiv preprint arXiv:1904.08900*, 2019.

[19] Y. Li, Y. Chen, N. Wang, and Z. Zhang. Scale-aware trident networks for object detection. *arXiv preprint arXiv:1901.01892*, 2019.

[20] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun. Light-head r-cnn: In defense of two-stage object detector. *arXiv preprint arXiv:1711.07264*, 2017.

[21] 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. doi: 10.1109/CVPR.2017.106. URL `doi.ieeecomputersociety.org/10.1109/CVPR.2017.106`.

[22] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[23] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[25] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 116–131, 2018.

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

[27] J. Redmon, S. Divvala, R. Girshick, and A. 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.

[28] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[29] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[30] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Lecun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, 2014.

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

[32] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[33] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018.

[34] X. Zhang, X. Zhou, M. Lin, and J. 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.

[35] X. Zhou, D. Wang, and P. Krahenbuhl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

[36] X. Zhou, J. Zhuo, and P. Krähenbühl. Bottom-up object detection by grouping extreme and center points. In *CVPR*, 2019.

[37] X. Zhu, H. Hu, S. Lin, and J. Dai. Deformable convnets v2: More deformable, better results. *CVPR*, 2019.

A More Results to Validate Hypothesis

In this section, we provide more results to validate our **hypothesis**:

> *The fully connected head is more suitable for object classification, while the convolution head is more suitable for object localization.*

To validate this hypothesis, we compare two variations of single head and four variations of double heads. The two variations of single head are described as follows:

Single-FC has a single fully connected head (*fc-head*), shared for both classification and bounding box regression. This is equivalent to the original FPN [21].

Single-Conv has a single convolution head (*conv-head*), shared for both classification and bounding box regression.

The four variations of double heads are described as follows:

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.

Note that for all six variations, we use FPN with ResNet-50 as backbone. *fc-head* has two fully connected layers (follow [21]) and *conv-head* stacks five residual blocks (details are shown in Section 3.2 in the submission draft). We trained a model for each variation on MS COCO 2014 *train+val* and evaluated on MS COCO 2014 *minival*.

The detection performances are shown in Table 5. 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*. This is corresponding to the hyper-parameter selection for Double-Head in Section 4.1 ($\omega^{fc} = 2.0$, $\omega^{conv} = 2.5$).

Double-Head outperforms detectors with single head 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 hypothesis that *fc-head* is more suitable for classification, while *conv-head* is more suitable for localization.

B More Examples for Qualitative Analysis

In this section, we provide more examples to compare the fully connected head (*fc-head*) with the convolution head (*conv-head*) qualitatively. 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 5 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 6 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 more accurate regression result for the same

	fc-head		*conv-head*							
	cls	reg	cls	reg	AP	$AP_{0.5}$	$AP_{0.75}$	AP_s	AP_m	AP_l
Single-FC	1.0	1.0			36.8	58.7	40.4	21.2	40.1	48.8
Single-Conv			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 5: Evaluations of single head and double heads on MS COCO 2014 *minival*, 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*. This is corresponding to the hyper-parameter selection for Double-Head in Section 4.1 in the submission draft ($\omega^{fc} = 2.0$, $\omega^{conv} = 2.5$). Clearly, Double-Head has the best performance, outperforming others by a non-negligible margin. Double-Head-Reverse has the worst performance.

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.

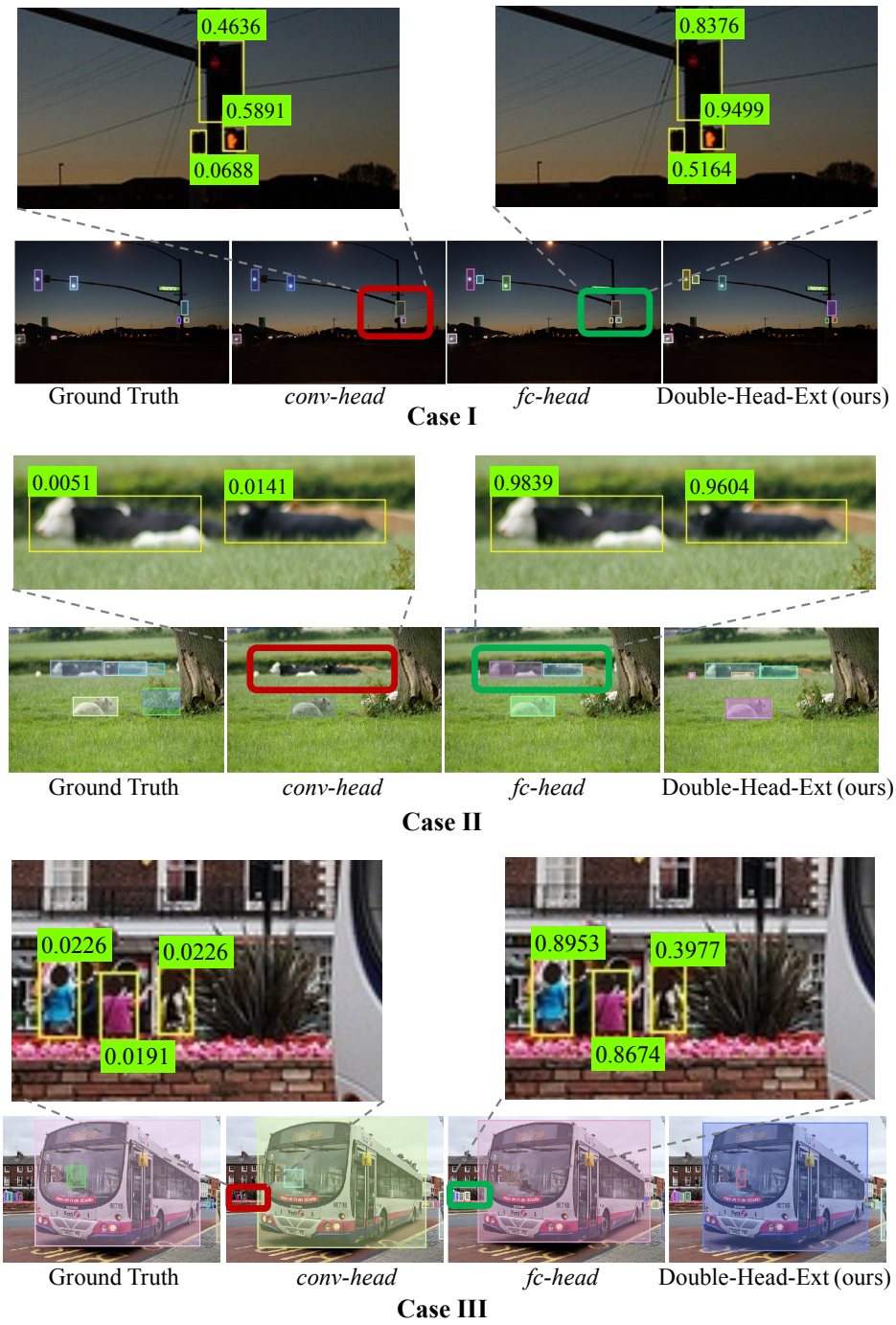


Figure 5: *fc-head* head is more suitable for classification than *conv-head*. This figure includes three cases. Each case has two rows. The bottom row shows ground truth, detection results using *conv-head* alone, *fc-head* alone, and our Double-Head-Ext (from left to right). The *conv-head* misses 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*.

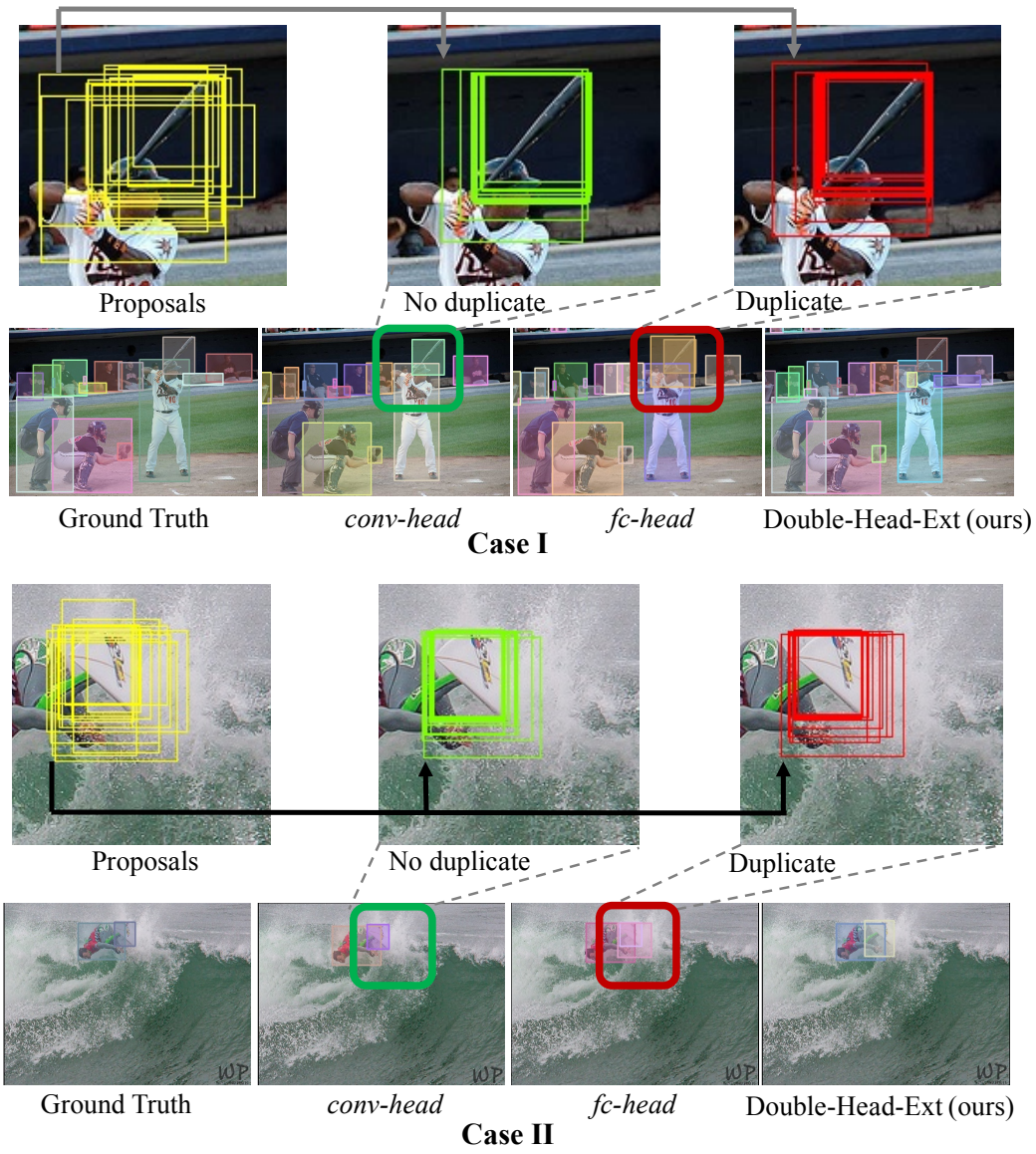


Figure 6: *conv-head* is more suitable for localization than *fc-head*. This figure includes two cases. Each case has two rows. The bottom row shows ground truth, detection results using *conv-head* alone, *fc-head* alone, and our Double-Head-Ext (from left to right). *fc-head* has a duplicate detection 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.