Weighted boxes fusion: Ensembling boxes from different object detection models

Roman Solovyev
Institute for Design Problems in Microelectronics of Russian Academy of Sciences
3, Sovetskaya Street, Moscow 124365, Russian Federation
roman.solovyev.zf@gmail.com

Weimin Wang
National University of Singapore
21 Lower Kent Ridge Rd, Singapore 119077
wangweimin777@gmail.com

Tatiana Gabruseva
Cork University Hospital
Cork, Ireland
tatigabru@gmail.com

Abstract

Object detection is a crucial task in computer vision systems with a wide range of applications in autonomous driving, medical imaging, retail, security, face recognition, robotics, and others. Nowadays, the neural networks-based models are used to localize and classify instances of objects of particular classes. When real-time inference is not required, the ensembles of models help to achieve better results.

In this work, we present a novel method for combining predictions of object detection models: weighted boxes fusion. Our algorithm utilizes confidence scores of all proposed bounding boxes to constructs the averaged boxes.

We tested method on several datasets and evaluated it in the context of the Open Images and COCO Object Detection tracks, achieving top results in these challenges. The 3D version of boxes fusion was successfully applied by the winning teams of Waymo Open Dataset and Lyft 3D Object Detection for Autonomous Vehicles challenges. The source code is publicly available at `https://github.com/ZFTurbo/Weighted-Boxes-Fusion`.

1. Introduction

Object detection is a computer vision technology that deals with detecting instances of semantic objects of a particular class in images and videos [35]. Detection is an essential task for a range of practical applications, including autonomous driving [16, 14], medical imaging [30, 15], robotics, security, and others. The task combines localization with classification. The object detection models typically return the proposed locations of the objects of a given class, class labels, and confidence scores.

The predicted boxes are selected using a non-maximum suppression (NMS) method [25]. First, it sorts all detection boxes by their confidence scores. Then, the detection box with the maximum confidence score is selected. At the same time, all other detection boxes with significant overlap to that box are filtered out. It relies on a hard-coded threshold to discard redundant boxes. Some recent works used a differentiable model to learn NMS and introduced soft-NMS [9] to improve filtering performance.

Such methods work well on a single model. However, they only select the boxes and can not produce averaged localization of predictions combined from various models effectively. Ensembles of models are widely used in applications that do not require real-time inference. Combining predictions from different models generalizes better and usually yields more accurate results compared to a single model [27]. The ensemble methods often get top places in machine learning competitions, for example, see [12, 15, 34, 33, 42, 7].

The other technique, commonly-used in practice, is a test-time augmentation (TTA). In this method, the predictions of the same model are obtained for the original and augmented images (for example, vertically/horizontally reflected) and then averaged.

In this paper, we propose a novel Weighted Boxes Fusion (WBF) method for combining predictions of object detection models. Unlike NMS and soft-NMS methods that simply remove part of the predictions, the proposed WBF method uses confidence scores of all proposed bounding boxes to constructs the average boxes.

This method significantly improves the quality of the combined predicted rectangles. The technique was evaluated in the context of the Open Images Object Detection Challenge and helped achieving one of the top results in the challenge [3]. The source code is publicly available at `https://github.com/ZFTurbo/`

`Weighted-Boxes-Fusion`.

The method can also be beneficial for the ensemble of the manual labels in the medical applications, i.e., combining labels of different expert radiologists in the pneumonia detection task. The averaged predictions of the expert panel should lead to more accurate labels. Hence it can produce better data for the computer-assisted diagnostics programs.

2. Related Work

2.1. Non-maximum suppression (NMS)

Predictions of a model in the image detection task consist of coordinates of the bounding box rectangle, a class label of the object, and a confidence score (probability from 0 to 1) that reflects how confident the model is in this prediction.

In the NMS method, the boxes are considered as belonging to a single object if their overlap, intersection-over-union (IoU) is higher than some threshold value. Thus, the boxes filtering process depends on selection of this single IoU threshold value, which affects the performance of the model. However, setting this threshold is tricky: if there are objects side by side, one of them would be eliminated. Figure 1 shows an example illustrating one such case. For the IoU threshold of 0.5, only one box prediction will remain. The other overlapping objects detected will be removed. Such errors reduce the precision of the model.

Figure 1: The photo illustrates several overlapping horses in the race. For several detections with the high confidence score, only one will be selected by the NMS algorithm for the IoU threshold above 0.5. Photo by Julia Joppien on Unsplash.

2.2. Soft-NMS

The soft-NMS method was proposed in [9] to reduce this problem. Instead of completely removing the detection proposals with high IoU and high confidence, it reduces the confidences of the proposals proportional to IoU value. On the example above, soft-NMS will lower the confidence scores proportionally to the IoU overlap. As the green detection box has a significant overlap with the yellow one, it will be removed. The recent adversarial attack has exploited this problem of the NMS and soft-NMS methods [41].

Soft-NMS demonstrated noticeable improvements over traditional NMS on standard benchmark datasets, like PASCAL VOC and MS COCO [5]. It has shown to improve performance for single models [9]. However, both NMS and soft-NMS discard redundant boxes, and thus can not produce averaged localization predictions from different models effectively.

3. Weighted Boxes Fusion

Here, we describe the novel method for the boxes fusion: Weighted Boxes Fusion (WBF). Suppose, we have bounding boxes predictions for the same image from $\mathbf{N}$ different models. Alternatively, we have $\mathbf{N}$ predictions of the same model for the original and augmented versions of the same image (i.e., vertically/horizontally reflected).

The WBF algorithm works in the following steps:

1. Each predicted box from each model is added to a single list $\mathbf{B}$. The list is sorted in decreasing order of the confidence scores $\mathbf{C}$.

2. Declare empty lists $\mathbf{L}$ and $\mathbf{F}$ for boxes clusters and fused boxes, respectively. Each position in the list $\mathbf{L}$ can contain a set of boxes (or single box), which form a cluster; each position in $\mathbf{F}$ contains only one box, which is the fused box from the corresponding cluster in $\mathbf{L}$. The equation to generate fused boxes will be discussed later.

3. Iterate through predicted boxes in $\mathbf{B}$ in a cycle and try to find a matching box in the list $\mathbf{F}$. The match is defined as a box with a large overlap with the box under question ($IoU > \mathbf{THR}$). Note: in our experiments, $\mathbf{THR} = 0.55$ was close to an optimal threshold.

4. If the match is not found, add the box from the list $\mathbf{B}$ to the end of lists $\mathbf{L}$ and $\mathbf{F}$ as new entries; proceed to the next box in the list $\mathbf{B}$.

5. If the match is found, add this box to the list $\mathbf{L}$ at the position $\mathbf{pos}$ corresponding to the matching box in the list $\mathbf{F}$.

6. Recalculate the box coordinates and confidence score in $\mathbf{F}[\mathbf{pos}]$, using all $\mathbf{T}$ boxes accumulated in cluster $\mathbf{L}[\mathbf{pos}]$, with the following fusion formulas:

$$\mathbf{C} = \frac{\sum_{i=1}^{\mathbf{T}} \mathbf{C}_i}{T}, \tag{1}$$

$$\mathbf{X1,2} = \frac{\sum_{i=1}^{\mathbf{T}} \mathbf{C}_i * \mathbf{X1,2}_i}{\sum_{i=1}^{\mathbf{T}} \mathbf{C}_i}, \quad (2)$$

$$\mathbf{Y1,2} = \frac{\sum_{i=1}^{\mathbf{T}} \mathbf{C}_i * \mathbf{Y1,2}_i}{\sum_{i=1}^{\mathbf{T}} \mathbf{C}_i}, \quad (3)$$

Note: we can use some nonlinear weights as well, for instance, $\mathbf{C^2}$, $sqrt(\mathbf{C})$, etc.

Set the confidence score for the fused box as the average confidence of all boxes that form it. Coordinates of the fused box are weighted sums of the coordinates of the boxes that form it, where the weights are confidence scores for the corresponding boxes. Thus, boxes with larger confidence contribute more to the fused box coordinates than boxes with lower confidence.

7. After all boxes in $\mathbf{B}$ are processed, re-scale confidence scores in $\mathbf{F}$ list: multiply it by a number of boxes in a cluster and divide by a number of models $\mathbf{N}$. If a number of boxes in the cluster is low, it could mean that only a small number of models predict it. Thus, we need to decrease confidence scores for such cases. It can be done in two ways:

$$\mathbf{C} = \mathbf{C} * \frac{min(T, N)}{N}, \quad (4)$$

or

$$\mathbf{C} = \mathbf{C} * \frac{T}{N}, \quad (5)$$

In practice, the outcome for both variants did not differ significantly, with the first being slightly better. In some cases, a number of boxes in one cluster can be more than a number of models.

Both NMS and soft-NMS exclude some boxes, while WBF uses all boxes. Thus, it can fix cases where all boxes are predicted inaccurately by all models. This case is illustrated in Figure3. NMS/soft-NMS will leave only one inaccurate box, while WBF will fuse it using all predicted boxes.

The non-maximum weighted (NMW) method proposed in [49, 26] has a similar idea. However, the NMW method does not change confidence scores; it uses IoU value to weight the boxes. NMW uses a box with the highest confidence to compare with, while WBF updates a fused box at each step, and uses it to check the overlap with the next predicted boxes. Also, NMW does not use information about how many models predict a given box in a cluster and, therefore, does not produce the best results for models' ensemble. We compare our proposed WBF method with NMW, NMS, and soft-NMS techniques in section 6.

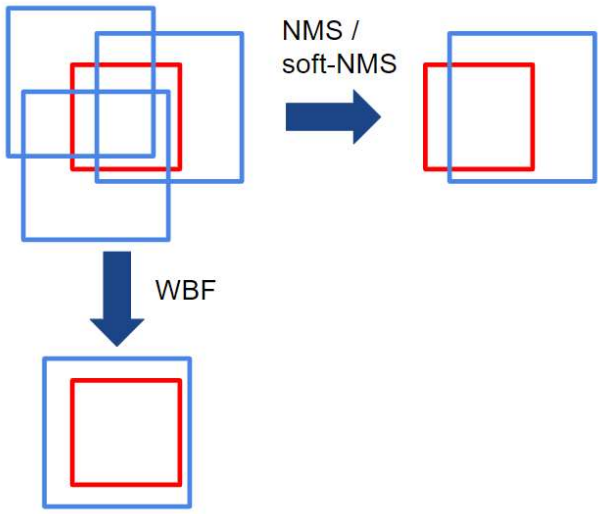

Figure 2: Schematic illustration of NMS/soft-NMS vs. WBF outcomes for an ensemble of inaccurate predictions. Blue – different models' predictions, red – ground truth.

4. Datasets

In this section, we describe data sets used to test the proposed WBF method of boxes fusion. We used the Open Images data set [20], the largest data set with object detection annotations to date, and Microsoft COCO [5, 23], one of the most popular benchmark datasets for object detection.

4.1. Open Images Dataset

The Open Images Object Detection Challenges held at the International Conference on Computer Vision 2019 and hosted on Kaggle [1]. The challenge used V5 release of the Open Images dataset [20] that includes around 16M bounding boxes for 600 object classes on 1.9M images. To date, it is the largest labeled dataset with object detection annotations.

In the Object Detection track of the challenge, the participants needed to predict a tight bounding box around object instances. The training set contained 12.2M bounding-boxes across 500 categories on 1.7M images. The boxes have mainly been manually drawn by professional annotators to ensure accuracy and consistency. The images of the dataset are very diverse and often contain complex scenes with several objects (7 per image on average). The details on the Open Images dataset are in [20].

The dataset consists of the training set (1743042 images), validation set (41620 images), and the test set (99999 images). The validation and test sets, as well as part of the training set, have human-verified image-level labels. As the test set labels are not available, we performed the ablation study on the validation set. The test set results were acces-

sible through submissions on the Kaggle web site.

4.2. COCO Dataset

The Microsoft COCO dataset became a popular benchmark for image detection and segmentation tasks [23]. The dataset contains over $200,000$ images with 80 object categories. Annotations for the training and validation sets (with over $500,000$ object instances) are publicly available here [5]. The models used in the ablation study presented in this paper were trained on the train2017 with 118k labeled images. Then, we tested the models on COCO validation set val2017 with 5K images. The best results were also submitted to the official leader board and evaluated on the test set with 20k images. [6].

5. Evaluation

For the Open Images Object Detection challenge, the evaluation metric was chosen by organizers. The models were evaluated using the mean average precision (mAP) at intersection-over-union (IoU) value equal to 0.5.

The IoU is a ratio of overlap between two objects (A and B) to the total area of the two objects combined. The IoU of a set of predicted bounding boxes (A) and ground truth bounding boxes (B) is calculated as:

$$IoU(A, B) = \frac{A \cap B}{A \cup B} \tag{6}$$

At each threshold value t, a precision value is calculated from the numbers of true positives (TP), false negatives (FN), and false positives (FP) as:

$$Precision(t) = \frac{TP(t)}{TP(t) + FP(t) + FN(t)} \tag{7}$$

For the Open Images Object Detection challenge, the AP is calculated at IoU = 0.5. The final mAP is computed as the average AP over the 500 classes in the challenge. The metric calculation is described on the Open Images Challenge website [2]. The implementation of this mAP variant is publicly available as part of the Tensorflow Object Detection API [4] under the name 'OID Challenge Object Detection Metric'.

For the COCO dataset, the mAP was computed for different intersection-over-union (IoU) thresholds. The metric sweeps over a range of IoU thresholds, at each point calculating an average precision value. The threshold values range from 0.5 to 0.95 with a step size of 0.05: (0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95). A predicted object is considered a "hit" if its intersection over union with a ground truth object is greater than 0.5.

The average precision (AP) of a single image is calculated as the mean of the above precision values at each IoU threshold:

$$AP = \frac{1}{|thresholds|} \sum_{t} \frac{TP(t)}{TP(t) + FP(t) + FN(t)} \tag{8}$$

The python implementation of this mAP metric variant can be found, for example, in [8].

6. Experiments

In this section, we perform an ablation study and compare the results obtained from WBF with NMS, soft-NMS, and the weighted NMW method. We also conduct specific experiments to understand when WBF gives the best boost to model performance. We used different datasets and models to make experiments more representative.

6.1. Models

A set of different detector models was used for the ablation study, including single-stage and two-stage detectors. Among single-shot detectors, we used RetinaNet [22] implementation in Keras with different ResNet [18] backbones, pre-trained on the Open Images Dataset [20]. For the COCO dataset, we used different versions of EfficientDet models [36] from [37] and DetectoRS model [28] from the official repository [29] with weights trained on MS COCO. All individual models produced predictions selected using the NMS algorithm.

The two-stage detectors used included Faster R-CNN [31], Mask R-CNN [17], and Cascade R-CNN [10].

6.2. An ensemble of two different models

Table 1 shows results for MS COCO validation set for two models (EfficientDetB6 and EfficientDetB7 [37] with NMS outputs) and their predictions ensemble. These models have recently shown top results on the COCO database [36]. The predictions were combined using four different techniques: NMS, soft-NMS, NMW, and WBF. We compare the resulting mAP(0.5...0.95), mAP(0.5) and mAP(0.75).

We used a grid search to find the optimal parameters for each of the four methods, which allowed achieving the best mAP for the given method. The individual models had different results on the MS COCO data set. Therefore, we also varied the weights for the model predictions. The optimal parameters are listed in Table 1. Even for the ensemble of only two models, the WBF method outperformed significantly other methods.

6.3. Test-time-augmentation ensemble

We tested the performance of different boxes combining methods for the TTA ensemble. We used the EfficientNetB7 model trained on the COCO dataset. We predict boxes for both original and horizontally mirrored images. Then, we

Model	mAP(0.5..0.95)	mAP(@0.5 IoU)	mAP(@0.5 IoU)
EffDetB6	0.513	0.705	0.554
EffDetB7	0.521	0.710	0.562
Method	mAP(0.5..0.95)	mAP(0.5)	mAP(0.75)
NMS	0.5269	0.7156	0.5737
IoU threshold	0.76	0.51	0.85
Weights	[3, 4]	[3, 4]	[2, 3]
Soft-NMS	0.5239	0.7121	0.5677
Threshold	0.0014	0.0013	0.0017
Sigma	0.15	0.145	0.16
Weights	[3, 4]	[3, 4]	[3, 4]
NMW	0.5285	0.7171	0.5743
IoU threshold	0.80	0.50	0.83
Weights	[3, 4]	[1, 1]	[3, 4]
WBF	**0.5344**	**0.7244**	**0.5824**
IoU threshold	0.75	0.59	0.77
Weights	[2, 3]	[3, 4]	[3, 4]

Table 1: Calculated mAP for individual EfficientDet models and ensembles of their predictions. The boxes were combined using the NMS, soft-NMS, NMW, and WBF methods. The optimal parameters are listed below for each method.

Model	mAP(0.5..0.95)	mAP(@0.5 IoU)	mAP(@0.5 IoU)
EffDetB7	0.521	0.710	0.562
EffDetB7 (Mirror)	0.519	0.710	0.558
Method	mAP(0.5..0.95)	mAP(0.5)	mAP(0.75)
NMS	0.5233	0.7129	0.5656
Soft-NMS	0.5210	0.7092	0.5633
NMW	0.5250	0.7138	0.5691
WBF	**0.5262**	**0.7144**	**0.5717**

Table 2: Calculated mAP for TTA ensemble of predictions from the EfficientDetB7 model. We combine predictions for original and horizontally mirrored images.

perform a grid search to find optimal parameters for all four ensemble methods. In this case, we used identical weights for both predictions. The results are in Table 2.

6.4. An ensemble of many different models

6.4.1 Ensemble of models for COCO Dataset

Here, we combined the results of several different models. We used several models trained on MS COCO models from the official EfficientDet repository [37], two DetectoRS [28, 29] models with two different backbones, and the YOLOv5 model [40]. For this experiment, we performed TTA and made predictions for both original and horizontally flipped images. The individual models' performance and the result for their ensemble are in Table 3. Each model used the NMS method for its output boxes prediction.

The weights in the final ensemble and IoU threshold for the WBF methods were optimized using the validation set. The WBF method for combining predictions of detection models gave **56.1** mAP for the validation data set and **56.4** mAP for the test set. It is a considerable improvement compared to the performance of individual models and gives the top third result on the MS COCO official leaderboard [6] (as of 07/08/2020). These results can be reproduced using a benchmark published on GitHub [46].

6.4.2 Ensemble of RetinaNet models for Open Images Dataset

The results obtained on the Open Images Dataset [20] for combining predictions of RetinaNet single-shot-detector models with different backbones are in Table 4. For the Open Images challenge, we used mAP at 0.5 IoU, suggested by the challenge organizers. Again, we used a grid search to find the optimal parameters for each of the methods. Combining predictions from several detectors with different backbones yields better performance, and the WBF method demonstrated the best results.

Results

#	User	Entries	Date of Last Entry	Team Name	AP ▲	AP IoU=.50 ▲	AP IoU=.75 ▲	AP large ▲	AP medium ▲	AP small ▲	AR large ▲	AR max=1 ▲	AR max=10 ▲	AR max=100 ▲	AR medium ▲	AR small ▲
1	JianhuaHan	19	06/11/20	Noah CV Lab	0.59 (1)	0.77 (2)	0.65 (1)	0.72 (1)	0.62 (1)	0.41 (1)	0.88 (2)	0.42 (1)	0.70 (1)	0.75 (1)	0.78 (1)	0.59 (1)
2	mmdet	4	10/04/19		0.58 (2)	0.77 (1)	0.64 (2)	0.71 (2)	0.60 (2)	0.40 (2)	0.86 (4)	0.41 (2)	0.69 (2)	0.74 (2)	0.77 (2)	0.58 (2)
3	ZFTurbo	6	07/14/20		0.56 (3)	0.74 (4)	0.63 (3)	0.70 (3)	0.60 (3)	0.38 (3)	0.86 (7)	0.41 (3)	0.67 (5)	0.72 (6)	0.76 (4)	0.55 (6)
4	zacurr	6	10/04/19	DeepAR(ETRIxKAIST_AIM)	0.55 (4)	0.75 (3)	0.61 (4)	0.67 (8)	0.58 (4)	0.38 (3)	0.86 (6)	0.40 (4)	0.67 (5)	0.72 (5)	0.76 (5)	0.56 (4)

Figure 3: The official leader boards of COCO Detection Challenge (Bounding Box) as of 07/08/20. The third top entry represents the results described in this paper. These results can be reproduced using benchmark published on GitHub [46].

Model	mAP @(0.5..0.95) original images	mAP @(0.5..0.95) flipped images
EfficientNetB4	49.0	48.8
EfficientNetB5	50.5	50.2
EfficientNetB6	51.3	51.1
EfficientNetB7	52.1	51.9
DetectoRS (X101)	51.5	51.5
DetectoRS (ResNet50)	49.6	49.6
YOLO v5x (TTA)	50.0	-
Ensemble	COCO validation	COCO test
WBF	**56.1**	**56.4**
IoU threshold	0.7	0.7
Weigths	[4,5,7,9,8,5,5]	[4,5,7,9,8,5,5]

Table 3: The mAP metrics for several individual models obtained on MS COCO validation set for the original and horizontally flipped images, and the resulting mAP metrics for the ensemble of their predictions.

RetinaNet backbone	mAP (@0.5 IoU)
ResNet152	0.5180
ResNet101	0.4997
ResNet50	0.4613
ResNet152 (v2)	0.5123
Method	mAP (@0.5 IoU)
NMS	0.5442
IoU threshold	0.3
Soft-NMS	0.5428
Sigma	0.05
WBF	**0.5665**
IoU threshold	0.55

Table 4: The mAP metrics for individual RetinaNet models and ensembles of their predictions. We used the same weights for each model.

6.4.3 Ensemble of fairly different models for Open Images Dataset

In this experiment, performed on the Open Images dataset, we used a range of different models. The ensemble included the RetinaNet models combination described above, Mask-RCNN with ResNext101 backbone [39], Cascade-RCNN with ResNext101 backbone, FPN, and GroupNorm [39], MMdetection HTC model [24], and Faster-RCNNs model with InceptionResNetV2 backbone from [38]. These models had a comparable performance on the Open Images dataset in terms of mAP. In the previous experiments, we studied the performance of the WBF method for similar models. Here, we explore combining predictions from highly different models.

The WBF method outperformed by far the other algorithms (see Table 5) and helped to achieve one of the top results (7 out of 558) in the challenge [3].

Model	Backbone	mAP (@0.5 IoU)
RetinaNet (Mix)	resnet50, resnet101, resnet152	0.5164
Mask-RCNN	ResNext101	0.5019
Cascade-RCNN	ResNext101-FPN-GroupNorm	0.5144
MMDetection HTC	X-101-64x4d-FPN	0.5152
Faster RCNN	InceptionResNetV2	0.4910
Method	Params	mAP (@0.5 IoU)
NMS	IoU threshold = 0.5	0.5642
Soft-NMS	Sigma 0.1, threshold 1e-03	0.5616
NMW	IoU threshold 0.5	0.5667
WBF	IoU threshold 0.6	**0.5982**

Table 5: Object detection models trained on Open Images dataset, the resulted mAPs for these models and for their ensembles obtained via NMS, soft-NMS, NMW, and WBF methods.

7. Discussion

Most neural networks for Object Detection use NMS or Soft-NMS output to filter predicted boxes. We tested our method for models' predictions that have already been processed by NMS. In this experiment, we wanted to check whether it is possible to use WBF at the output of a model instead of NMS. For the experiment, we used the RetinaNet [22] detector with the ResNet152 [18] backbone trained on the Open Images dataset. After we disabled the last NMS layer, we got 500 boxes with their probability scores. We compared the mAP (@0.5 IoU) metric for combining these output predictions via NMS and WBF algorithms.

The results are the following:

1. Raw boxes (no NMS/WBF) - mAP: 0.1718 - the value is rather small because of many overlapping boxes
2. NMS with default IoU threshold = 0.5 (e.g. standard model output) – mAP: 0.4902
3. NMS with optimal IoU threshold = 0.47 – mAP: 0.4906 – the tiny change from the default threshold
4. WBF with optimal parameters – mAP: 0.4532 (the optimized parameters: IoU threshold = 0.43, skip threshold = 0.21)

As can be seen from the experiment, replacing the NMS method at the output of the individual model with WBF results in the degradation of the model performance. We think it is due to the excessive number of low scored wrong predictions given by the model output. The NMS or soft-NMS produces more efficient filtering for those predictions, while the WBF works better when used for the models' ensemble.

Thus, WBF works well for combining boxes for fairly accurate models. However, when it comes to a large number of overlapping boxes with different confidence scores, WBF gives worse results than NMS.

The method was expanded on the 3D boxes fusion as well. The code for the 3D boxes fusion is available here: [47]. In the recent Waymo Open Dataset Challenge [43], the winners from the first place of 3D Detection and Domain Adaptation track [13] and the second place from the 2D Object Detection track [11] used this method for combining predictions. The first place winners from Lyft 3D Object Detection for Autonomous Vehicles [19] reported that WBF gives better score for their ensemble (0.222 vs 0.220) [48]. It was also reported that the WBF method gives the best results on the Wheat detection dataset for boxes ensemble comparing to other methods [32].

WBF is currently slower than the NMS and soft-NMS methods. While the speed depends on the set of boxes, a number of classes, and software implementation, the WBF algorithm, on average, is around three times slower than the standard NMS.

8. Conclusion

In this work, we proposed a new technique for combining predictions of object detection models, both for 2D and 3D boxes. The described WBF method uses confidence scores of all proposed bounding boxes in the iterative algorithm that constructs the averaged boxes. We evaluated our method and compared its performance to the alternatives on two popular benchmark datasets for object detection: the Open Images [20] and Microsoft COCO [23] datasets.

The WBF method outperformed by far other results for combining predictions. It helped achieving one of the top results in the Open Images Detection Challenge and the top performance for the COCO Detection Challenge ((**56.1** mAP for validation data set and **56.4** mAP for the test-dev set). It's a considerable improvement compared to the performance of individual models.

The method is also available for the 3D boxes fusion. The 3D version of the WBF algorithm was successfully applied by the winners of the Waymo Detection challenge [13,

11] and Lyft 3D Object Detection for Autonomous Vehicles challenge [19, 48]. The source code for the WBF and the usage examples is available at GitHub: https://github.com/ZFTurbo/Weighted-Boxes-Fusion.

References

[1] Open images object detection challenge. https://www.kaggle.com/c/open-images-2019-object-detection/overview, 2019.

[2] Open images object detection challenge, evaluation. https://storage.googleapis.com/openimages/web/evaluation.html#object_detection_eval, 2019.

[3] Open images object detection challenge, leaderboard. https://www.kaggle.com/c/open-images-2019-object-detection/leaderboard, 2019.

[4] Tensorflow object detection api. https://github.com/tensorflow/models/tree/master/research/object_detection, 2019.

[5] Coco dataset. cocodataset, 2020.

[6] Coco detection challenge (bounding box) leaderboard. leaderboard, 2020.

[7] James Atwood, Yoni Halpern, Pallavi Baljekar, Eric Breck, D Sculley, Pavel Ostyakov, Sergey I Nikolenko, Igor Ivanov, Roman Solovyev, Weimin Wang, et al. The inclusive images competition. In *The NeurIPS'18 Competition*, pages 155–186. Springer, 2020.

[8] Sergei Belousov. map: Mean average precision for object detection. https://github.com/bes-dev/mean_average_precision, 2020.

[9] 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.

[10] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018.

[11] Sijia Chen, Yu Wang, Li Huang, Runzhou Ge, Yihan Hu, Zhuangzhuang Ding, and Jie Liao. 2nd place solution for waymo open dataset challenge – 2d object detection. *arXiv*, 2020.

[12] David. Statoil/c-core iceberg classifier challenge. 1st place solution overview. https://www.kaggle.com/c/statoil-iceberg-classifier-challenge/discussion/48241, 2018.

[13] Zhuangzhuang Ding, Yihan Hu, Runzhou Ge, Li Huang, Sijia Chen, Yu Wang, and Jie Liao. 1st place solution for waymo open dataset challenge – 3d detection and domain adaptation. *arXiv*, 2020.

[14] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: A benchmark. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, jun 2009.

[15] Tatiana Gabruseva, Dmytro Poplavskiy, and Alexandr A. Kalinin. Deep learning for automatic pneumonia detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

[16] A Geiger, P Lenz, C Stiller, and R Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, aug 2013.

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

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[19] Kaggle. Lyft 3d object detection for autonomous vehicles. https://www.kaggle.com/c/3d-object-detection-for-autonomous-vehicles/leaderboard, 2019.

[20] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*, 2018.

[21] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv:1811.00982*, 2018.

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[23] 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 *Computer Vision – ECCV 2014*, pages 740–755. Springer International Publishing, 2014.

[24] MMDetection. Github: Mmdetection benchmark and model zoo. https://github.com/open-mmlab/mmdetection/blob/master/docs/MODEL_ZOO.md, 2019.

[25] A Neubeck and L Van Gool. Efficient non-maximum suppression. In *Proceedings of the International Conference on Pattern Recognition*, volume 3, page 850–855, 2006.

[26] Chengcheng Ning, Huajun Zhou, Yan Song, and Jinhui Tang. Inception single shot MultiBox detector for object detection. In *2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*. IEEE, jul 2017.

[27] Oleg Okun, Giorgio Valentini, and Matteo Re. *Ensembles in machine learning applications*, volume 373. Springer Science & Business Media, 2011.

[28] Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. *arXiv preprint arXiv:2006.02334*, 2020.

[29] Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Github: Detectors. https://github.com/joe-siyuan-qiao/DetectoRS, 2020.

[30] Pranav Rajpurkar, Jeremy Irvin, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Ding, Aarti Bagul, Curtis Langlotz, Katie Shpanskaya, Matthew P. Lungren, and Andrew Y. Ng. Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. *arXiv:1711.05225v1*, 2017.

[31] 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.

[32] Alex Shonenkov. Bayesian optimization wbf for efficientdet. https://www.kaggle.com/shonenkov/bayesian-optimization-wbf-efficientdet, 2020.

[33] Roman Solovyev. Roof material classification from aerial imagery. *arXiv preprint arXiv:2004.11482*, 2020.

[34] Roman A Solovyev, Maxim Vakhrushev, Alexander Radionov, Irina I Romanova, Aleksandr A Amerikanov, Vladimir Aliev, and Alexey A Shvets. Deep learning approaches for understanding simple speech commands. In *2020 IEEE 40th International Conference on Electronics and Nanotechnology (ELNANO)*, pages 688–693. IEEE, 2020.

[35] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer-Verlag, Berlin, Heidelberg, 1st edition, 2010.

[36] Mingxing Tan, Ruoming Pang, and Quoc V. Le. Efficientdet: Scalable and efficient object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020)*, 2019.

[37] Mingxing Tan, Ruoming Pang, and Quoc V. Le. Github: Efficientdet. https://github.com/google/automl/tree/master/efficientdet, 2020.

[38] Tensorflow. Github: Tensorflow detection model zoo. https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md, 2019.

[39] Tensorpack. Github: Tensorpack faster r-cnn. https://github.com/tensorpack/tensorpack/tree/master/examples/FasterRCNN, 2019.

[40] ultralytics. Yolo v5 on github. https://github.com/ultralytics/yolov5, 2020.

[41] Derui Wang, Chaoran Li, Sheng Wen, Qing-Long Han, Surya Nepal, Xiangyu Zhang, and Yang Xiang. Daedalus: Breaking non-maximum suppression in object detection via adversarial examples. *arXiv preprint arXiv:1902.02067*, 2020.

[42] W Wang, RA Solovyev, AL Stempkovsky, DV Telpukhov, and AA Volkov. Method for whale re-identification based on siamese nets and adversarial training. *Optical Memory and Neural Networks*, 29(2):118–132, 2020.

[43] Waymo. Waymo open dataset challenge. https://waymo.com/open/challenges, 2020.

[44] ZFTurbo. Keras-retinanet for open images challenge 2018. https://github.com/ZFTurbo/Keras-RetinaNet-for-Open-Images-Challenge-2018, 2018.

[45] ZFTurbo. Planet: Understanding the amazon from space. 3rd place solution. https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/discussion/38831, 2018.

[46] ZFTurbo. Coco wbf benchmark. https://github.com/ZFTurbo/Weighted-Boxes-Fusion/tree/master/benchmark, 2020.

[47] ZFTurbo. Wbf for 3d boxes. https://github.com/ZFTurbo/Weighted-Boxes-Fusion/blob/master/ensemble_boxes/ensemble_boxes_wbf_3d.py, 2020.

[48] Wenjing Zhang. Lyft 3d object detection for autonomous vehicles. https://www.kaggle.com/c/3d-object-detection-for-autonomous-vehicles/discussion/122820, 2020.

[49] Huajun Zhou, Zechao Li, Chengcheng Ning, and Jinhui Tang. Cad: Scale invariant framework for real-time object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 760–768, 2017.