Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

Weighted boxes fusion: Ensembling boxes from different object detection models

Roman Solovyev [a],*, Weimin Wang [b], Tatiana Gabruseva [c]

[a] Institute for Design Problems in Microelectronics of Russian Academy of Sciences, 3, Sovetskaya Street, Moscow 124365, Russian Federation
[b] National University of Singapore, 21 Lower Kent Ridge Rd, 119077, Singapore
[c] Cork University Hospital, Cork, Ireland

A R T I C L E   I N F O

Article history:
Received 26 November 2020
Received in revised form 12 January 2021
Accepted 14 January 2021
Available online 3 February 2021

Keywords:
Object detection
Computer vision
Deep learning

A B S T R A C T

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, neural networks-based models are used to localize and classify instances of objects of particular classes. When real-time inference is not required, ensembles of models help to achieve better results.
In this work, we present a novel method for fusing predictions from different object detection models: weighted boxes fusion. Our algorithm utilizes confidence scores of all proposed bounding boxes to construct averaged boxes.
We tested the method on several datasets and evaluated it in the context of Open Images and COCO Object Detection challenges, 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 GitHub (Solovyev, 2019 [31]).
We present a novel method for combining predictions in ensembles of different object detection models: weighted boxes fusion. This method significantly improves the quality of the fused predicted rectangles for an ensemble.
We tested the method on several datasets and evaluated it in the context of the Open Images and COCO Object Detection challenges. It helped to achieve top results in these challenges. The source code is publicly available at GitHub.

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 [8,10], medical imaging [9,27], robotics, security, and others. This task combines localization with classification. Object detection models typically return proposed locations of the objects of a given class, class labels, and confidence scores.

Predicted boxes are selected using a non-maximum suppression (NMS) method [22]. First, it sorts all detection boxes by their confidence scores. Then, the detected box with the maximum confidence score is selected. At the same time, all other detection boxes with significant overlap to that selected 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 [3] to improve filtering performance.

These methods work well on a single model. However, they only select boxes and can not produce averaged localization of predictions combined from various models effectively. This makes them less efficient for ensembles of different models. Ensembles 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 [24]. The ensemble methods often get top performance in machine learning competitions, for example, see [1,9,30,34].

In this paper, we propose a novel Weighted Boxes Fusion (WBF) method for fusing predictions of different object detection models. Unlike NMS and soft-NMS methods that simply remove some predictions, the proposed WBF method uses confidence scores of all proposed bounding boxes to constructs average boxes. The WBF method works well for combining predictions of several models after applying the traditional NMS method on each of them separately. As we show in Discussion it gives worse results when applied to single model predictions instead of NMS. We show the experiments in Section 6.

For model ensembles, this method significantly improves the quality of fused predicted rectangles. The technique was evaluated in the

* Corresponding author.
E-mail address: roman.solovyev.zf@gmail.com (R. Solovyev).

https://doi.org/10.1016/j.imavis.2021.104117
0262-8856/© 2021 Published by Elsevier B.V.

context of Open Images and COCO Object Detection challenges and helped to achieve one of the top results in the challenge [17]. The source code is publicly available at GitHub [31].

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 Section 6 we show the WBF method applications for TTA ensembles.

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

2. Related work

2.1. Non-maximum suppression (NMS)

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

In the NMS method, 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 the 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. Fig. 1 shows an example illustrating one such case. For the IoU threshold of 0.5, only one box prediction will remain. The other overlapping detected boxes will be removed. Such errors reduce the precision of the model.

2.2. Soft-NMS

The soft-NMS method was proposed in [3] to reduce this problem. Instead of completely removing the detection proposals with high IoU and high confidence, it reduces the confidence of proposals proportional to the 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 box, it will be removed. The recent adversarial attack has exploited this problem of NMS and soft-NMS methods [42].

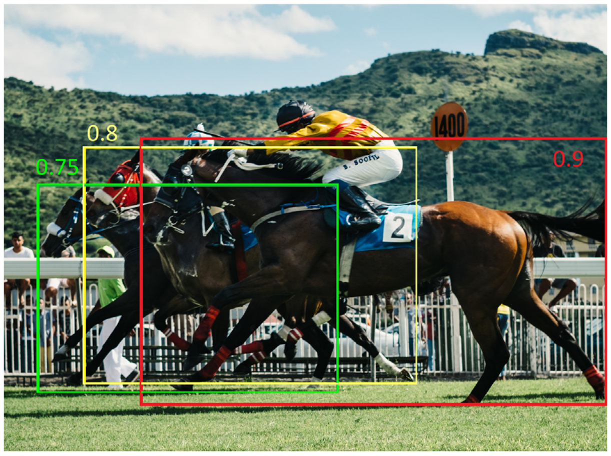


Fig. 1. The photo illustrates several overlapping horses in a race. For several detections with high confidence scores, only one will be selected by the NMS algorithm for the IoU threshold above 0.5. The other overlapping detected boxes will be removed and yield false negatives. Photo by Julia Joppien on Unsplash.

Soft-NMS demonstrated noticeable improvements over the traditional NMS on standard benchmark datasets, like PASCAL VOC and MS COCO [13]. It has been shown to improve performance for single models [3]. However, both NMS and soft-NMS discard redundant boxes, and thus can not produce averaged localization predictions combined from different models effectively.

3. Weighted boxes fusion

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

WBF algorithm works in the following steps:

1. Each predicted box from each model is added to a single list **B**. The list is sorted in decreasing order of confidence scores **C**.
2. Declare empty lists **L** and **F** for boxes clusters and fused boxes, respectively. Each position in the list **L** can contain a set of boxes (or single box), which form a cluster. Each position in **F** contains only one box, which is the fused box from the corresponding cluster in **L**. The equation to generate fused boxes will be discussed later.
3. Iterate through predicted boxes in **B** in a cycle and try to find a matching box in the list **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 **B** to the end of lists **L** and **F** as new entries; proceed to the next box in the list **B**.
5. If the match is found, add this box to the list **L** at the position **pos** corresponding to the matching box in the list **F**.
6. Recalculate box coordinates and its confidence score in **F**[**pos**], using all **T** boxes accumulated in the cluster **L**[**pos**], with the following fusion formulas:

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

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

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

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 boxes that form it, where 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 **B** are processed, re-scale confidence scores in **F** list: multiply it by a number of boxes in a cluster and divide by a number of models **N**. If the 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:

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

or

$$\mathbf{C} = \mathbf{C} * \frac{T}{N}, \tag{5}$$

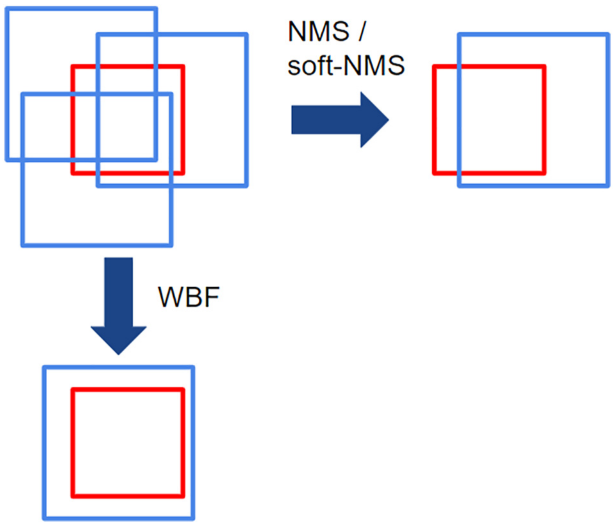


Fig. 2. Schematic illustration of NMS/soft-NMS vs. WBF outcomes for an ensemble of inaccurate predictions. Blue – different models' predictions, red – ground truth.

In practice, the outcome for both variants does not differ significantly, with the first being slightly better. In some cases, the number of boxes in one cluster can be more than the 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 Fig. 2. 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 [23,45] has a similar idea. However, the NMW method does not change confidence scores; it uses the 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.

4. Datasets

In this section, we describe data sets used to test the proposed WBF method for fusing boxes. We used the Open Images data set [18], the largest data set with object detection annotations to date, and Microsoft COCO [20], one of the most popular benchmark datasets for object detection.

4.1. Open images dataset

Open Images Object Detection Challenges were held at the International Conference on Computer Vision 2019 and hosted on Kaggle [15]. The challenge used V5 release of the Open Images dataset [18] that includes around 16 M bounding boxes for 600 object classes on 1.9 M images. To date, it is the largest labeled dataset with object detection annotations.

In the Object Detection challenge, participants needed to predict a tight bounding box around object instances. The training set contained 12.2 M bounding-boxes across 500 categories on 1.7 M images. Bounding 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 [18].

The dataset consists of the training set (1,743,042 images), validation set (41,620 images), and the test set (99,999 images). The validation and test sets, as well as part of the training set, have human-verified image-level labels. As test set labels are not available, we performed the ablation study on the validation set. The test set results were accessible through submissions on the Kaggle website.

4.2. COCO dataset

The Microsoft COCO dataset became a popular benchmark for image detection and segmentation tasks [20]. 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 [13]. The models used in the ablation study presented in this paper were trained on the train2017 with 118 k labeled images. Then, we tested the models on COCO validation set val2017 with 5 K images. The best results were also submitted to the official leader board and evaluated on the test set with 20 k images [6].

5. Evaluation

For the Open Images Object Detection challenge, the evaluation metric was chosen by organizers. Models were evaluated using 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 a total area of 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 average precision 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 [16]. The implementation of this *mAP* variant is publicly available as part of the Tensorflow Object Detection API [39] under the name'OID Challenge Object Detection Metric'.

For the COCO dataset, 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. 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 a mean of the 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 [2].

6. Experiments

In this section, we perform an ablation study and compare results obtained from models ensembling using WBF, NMS, soft-NMS, and

NMW methods. We also conduct specific experiments to understand when WBF gives the best boost to ensemble performance. We used different datasets and models to make experiments more representative.

6.1. Models

We used a set of different detectors models for the ablation study, including single-stage and two-stage detectors. Among single-stage detectors, we used RetinaNet [19] implementation in Keras [29] with different ResNet [12] backbones, pre-trained on the Open Images Dataset [18]. For the COCO dataset, we used different versions of EfficientDet models [36] from [37] and DetectoRS model [25] from the official repository [26] with weights trained on MS COCO. All individual models produced predictions selected using NMS algorithm.

The two-stage detectors used included Faster R-CNN [28], Mask R-CNN [11], and Cascade R-CNN [4].

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..0.0.95), 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 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.

Table 1
Calculated mAP for individual EfficientDet models and ensembles of their predictions. Predicted boxes from two models were combined using NMS, soft-NMS, NMW, and WBF methods. Optimal parameters are listed below for each method. Before ensembling, the outputs of individual models were processed using the NMS method.

Model	mAP(0.5.0.0.95)	mAP(@0.5 IoU)	mAP(@0.75 IoU)
EffDetB6	0.513	0.705	0.554
EffDetB7	0.521	0.710	0.562
Method	mAP(0.5.0.0.95)	mAP(@0.5 IoU)	mAP(@0.75 IoU)
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]	[3, 4]
WBF	**0.5344**	**0.7244**	**0.5824**
IoU threshold	0.75	0.59	0.77
Skip box threshold	10^{-7}	10^{-7}	10^{-7}
Weights	[2, 3]	[3, 4]	[3, 4]

Table 2
Calculated mAP for TTA ensemble of predictions from the EfficientDetB7 model. We combine predictions for original and horizontally mirrored images. Optimal parameters for each method were found via a grid search.

Model	mAP(0.5.0.0.95)	mAP(@0.5 IoU)	mAP(@0.75 IoU)
EffDetB7	0.521	0.710	0.562
EffDetB7 (Mirror)	0.519	0.710	0.558
Method	mAP(0.5.0.0.95)	mAP(@0.5 IoU)	mAP(@0.75 IoU)
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**

6.4. An ensemble of many different models

6.4.1. Ensemble of models for COCO dataset

Here, we combine predictions from several different models. We used models trained on MS COCO models from the official EfficientDet repository [37], DetectoRS [25,26] models with two different backbones, and YOLOv5 model [41]. For this experiment, we performed TTA and made predictions for both original and horizontally flipped images. The individual models' performance and results for their ensemble are in Table 3. Each individual model used the NMS method for filtering its output prediction.

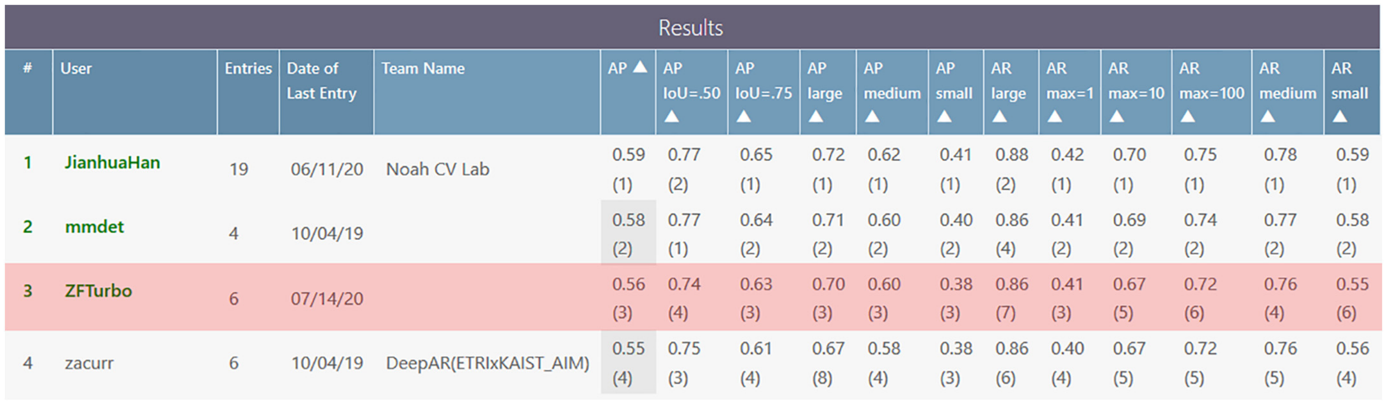
Weights in the final ensemble and IoU threshold for the WBF method were optimized using the validation set. The WBF method for fusing predictions of detection models gave **56.1** mAP for the validation data set and **56.4** mAP for the test set. It is a considerable improvement compared to the performance of individual models and gives the top third result on the MS COCO official leaderboard [6] (as of 07/08/2020) see Fig. 3. These results can be reproduced using a benchmark published on GitHub [32].

6.4.2. Ensemble of RetinaNet models for open images dataset

The results obtained on Open Images Dataset [18] for combining predictions from RetinaNet models with different backbones are shown 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 optimal parameters for each method. Combining predictions from several detectors with different backbones yields better performance, and the WBF method demonstrated the best results for the ensemble of different models. Before ensembling, outputs of individual models were selected using the NMS algorithm.

Table 3
The mAP metrics for individual models obtained on MS COCO validation set for original and horizontally flipped images; and mAP metrics for ensembles of their predictions made using different methods. Before ensembling, outputs of individual models were filtered using the NMS algorithm.

Model	mAP @(0.5.0.0.95)	mAP @(0.5.0.0.95)
	original images	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 v5	50.0	50.0
Ensemble	COCO validation	COCO test
NMS	53.4	53.8
IoU threshold	0.72	0.72
Weights	[2,4,7,9,8,2,2]	[2,4,7,9,8,2,2]
WBF	**56.1**	**56.4**
IoU threshold	0.70	0.70
Skip box threshold	10^{-3}	10^{-3}
Weigths	[4,5,7,9,8,5,5]	[4,5,7,9,8,5,5]



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)

Fig. 3. The official leader board 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 [32].

Table 4
The mAP metrics for individual RetinaNet models with NMS, and ensembles of their predictions using different methods. We used equal weights for all models. Before ensembling, outputs of individual models were filtered using the NMS algorithm.

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
Skip box threshold	10^{-7}

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 [40], Cascade-RCNN with ResNext101 backbone, FPN, and GroupNorm [40], MMdetection HTC model [21], and Faster-RCNNs model with InceptionResNetV2 backbone from [38]. These models had a comparable mAP metrics on the Open Images dataset. In the previous experiments, we studied the performance of the WBF method for similar models. Here, we explore combining predictions from highly different models.

Table 5
Models trained on the Open Images dataset and their individual mAPs metrics; mAP metrics obtained for all models ensembles via the NMS, soft-NMS, NMW, and WBF methods. Before ensembling, outputs of individual models were filtered using the NMS algorithm. All models' weights were equal.

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 0.001	0.5616
NMW	IoU threshold 0.5	0.5667
WBF	IoU threshold 0.6, skip box thr 10^{-7}	**0.5982**

As shown in Table 5 the WBF method outperformed by far the other algorithms and helped to achieve one of the top results (7 out of 558) in the challenge [17].

7. Discussion

Most neural networks for Object Detection use NMS or Soft-NMS output to filter predicted boxes. Previously, we tested our method for models' predictions that have already been filtered 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 [19,29] detector with the ResNet152 [12] 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 an individual model with the WBF method results in degradation of the model performance. We think it is due to the excessive number of low scored wrong predictions given by a single model output. NMS or soft-NMS produce more efficient filtering for those predictions, while 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 is available for a fusion of 3D boxes [33]. In the recent Waymo Open Dataset Challenge [43], the winners of the 3D Detection and Domain Adaptation challenge [7] and the second place from the 2D Object Detection challenge [5] used the WBF method for combining predictions. The winners of Lyft 3D Object Detection for Autonomous Vehicles competition [14] reported that WBF gives a better score for their ensemble (0.222 vs 0.220) [44].

The 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 [18] and Microsoft COCO [20] datasets.

The WBF method outperformed by far other results for combining predictions of different models. It helped to achieve top results in the Open Images and COCO Detection Challenges ((**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. However, for the output of a single model, it gives worse results than the NMS due to excessive number of low scored wrong predictions. The NMS or soft-NMS methods produce more efficient filtering of those predictions. The WBF works better for fusing predictions of different models, that have been pre-processed via NMS before adding them to an ensemble.

The WBF method is available for a fusion of 3D boxes as well [33]. The 3D version of the WBF algorithm was successfully applied by the winners of the Waymo Detection challenge [5,7] and Lyft 3D Object Detection for Autonomous Vehicles challenge [14]. The source code for WBF and usage examples is available at GitHub [31].

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] J. Atwood, Y. Halpern, P. Baljekar, E. Breck, D. Sculley, P. Ostyakov, S.I. Nikolenko, I. Ivanov, et al., The inclusive images competition, The NeurIPS'18 Competition, Springer 2020, pp. 155–186.
[2] S. Belousov, Map: Mean Average Precision for Object Detection, URL: https://github.com/bes-dev/mean_average_precision 2020.
[3] N. Bodla, B. Singh, R. Chellappa, L.S. Davis, Soft-nms–improving object detection with one line of code, Proceedings of the IEEE International Conference on Computer Vision 2017, pp. 5561–5569.
[4] Z. Cai, N. Vasconcelos, Cascade r-cnn: delving into high quality object detection, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2018, pp. 6154–6162.
[5] S. Chen, Y. Wang, L. Huang, R. Ge, Y. Hu, Z. Ding, J. Liao, 2nd place solution for waymo open dataset challenge – 2d object detection, arXiv (2020) 1–5 arXiv:2006.15507v1.
[6] Codalab, Coco Detection Challenge (Bounding Box) Leaderboard, URL: https://competitions.codalab.org/competitions/20794#leaderboard 2020.
[7] Z. Ding, Y. Hu, R. Ge, L. Huang, S. Chen, Y. Wang, J. Liao, 1st place solution for waymo open dataset challenge – 3d detection and domain adaptation, arXiv (2020) arXiv:2006.15505v1.
[8] P. Dollar, C. Wojek, B. Schiele, P. Perona, Pedestrian detection: a benchmark, 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009https://doi.org/10.1109/cvpr.2009.5206631.
[9] T. Gabruseva, D. Poplavskiy, A.A. Kalinin, Deep learning for automatic pneumonia detection, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2020 , URL: https://arxiv.org/abs/2005.13899.
[10] A. Geiger, P. Lenz, C. Stiller, R. Urtasun, Vision meets robotics: the KITTI dataset, Int. J. Robot. Res. 32 (2013) 1231–1237, https://doi.org/10.1177/0278364913491297.
[11] K. He, G. Gkioxari, R P D. R., G, Mask r-cnn, IEEE International Conference on Computer Vision 2017, pp. 2980–2988.
[12] K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image Recognition, URL: https://arxiv.org/abs/1512.03385 2015.
[13] Coco Dataset, URL: https://cocodataset.org/#cocodataset 2020.
[14] Kaggle, Lyft 3d Object Detection for Autonomous Vehicles, URL: https://www.kaggle.com/c/3d-object-detection-for-autonomous-vehicles/leaderboard 2019.
[15] Kaggle, Open Images Object Detection Challenge, URL: https://www.kaggle.com/c/open-images-2019-object-detection/overview 2019.
[16] Kaggle, Open Images Object Detection Challenge, Evaluation, URL: https://storage.googleapis.com/openimages/web/evaluation.html 2019.
[17] Kaggle, Open Images Object Detection Challenge, Leaderboard, URL: https://www.kaggle.com/c/open-images-2019-object-detection/leaderboard 2019.
[18] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, T. Duerig, et al., The open images dataset v4: unified image classification, object detection, and visual relationship detection at scale, arXiv (2018) preprint arXiv:1811.00982.
[19] T.Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, Proceedings of the IEEE International Conference on Computer Vision 2017, pp. 2980–2988.
[20] T.Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft COCO: common objects in context, Computer Vision – ECCV 2014, Springer International Publishing 2014, pp. 740–755, https://doi.org/10.1007/978-3-319-10602-1_48.
[21] MMDetection, Github: Mmdetection Benchmark and Model Zoo, URL: https://github.com/open-mmlab/mmdetection/blob/master/docs/MODEL_ZOO.md 2019.
[22] A. Neubeck, L. Van Gool, Efficient non-maximum suppression, Proceedings of the International Conference on Pattern Recognition 2006, pp. 850–855.
[23] C. Ning, H. Zhou, Y. Song, J. Tang, Inception single shot MultiBox detector for object detection, 2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), IEEE, 2017https://doi.org/10.1109/icmew.2017.8026312.
[24] O. Okun, G. Valentini, M. Re, Ensembles in Machine Learning Applications, 373, Springer Science & Business Media, 2011.
[25] S. Qiao, L.C. Chen, A. Yuille, Detectors: detecting objects with recursive feature pyramid and switchable atrous convolution, arXiv (2020) preprint arXiv:2006.02334 arXiv:2006.02334v1.
[26] S. Qiao, L.C. Chen, A. Yuille, Github: Detectors, URL: https://github.com/joe-siyuan-qiao/DetectoRS 2020.
[27] P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya, M.P. Lungren, A.Y. Ng, Chexnet: radiologist-level pneumonia detection on chest x-rays with deep learning, arXiv (2017) 1711.05225v1.
[28] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: towards real-time object detection with region proposal networks, Advances in Neural Information Processing Systems 2015, pp. 91–99.
[29] R. Solovyev, Keras-Retinanet for Open Images Challenge 2018, URL: https://github.com/ZFTurbo/Keras-RetinaNet-for-Open-Images-Challenge-2018 2018.
[30] R. Solovyev, Planet: Understanding the Amazon from Space. 3rd Place Solution, URL: https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/discussion/38831 2018.
[31] R. Solovyev, Github: Weighted Boxes Fusion, URL: https://github.com/ZFTurbo/Weighted-Boxes-Fusion 2019.
[32] R. Solovyev, Coco Wbf Benchmark, URL: https://github.com/ZFTurbo/Weighted-Boxes-Fusion/tree/master/benchmark 2020.
[33] R. Solovyev, Wbf for 3d Boxes, URL: https://github.com/ZFTurbo/Weighted-Boxes-Fusion/blob/master/ensemble_boxes/ensemble_boxes_wbf_3d.py 2020.
[34] R.A. Solovyev, M. Vakhrushev, A. Radionov, I.I. Romanova, A.A. Amerikanov, V. Aliev, A.A. Shvets, Deep learning approaches for understanding simple speech commands, 2020 IEEE 40th international conference on electronics and nanotechnology (ELNANO), IEEE 2020, pp. 688–693.
[35] R. Szeliski, Computer Vision: Algorithms and Applications, 1st ed. Springer-Verlag, Berlin, Heidelberg, 2010.
[36] M. Tan, R. Pang, Q.V. Le, Efficientdet: scalable and efficient object detection, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020) arXiv:1911.09070v6, 2019.
[37] M. Tan, R. Pang, Q.V. Le, Github: Efficientdet, URL: https://github.com/google/automl/tree/master/efficientdet 2020.
[38] Tensorflow, Github: Tensorflow Detection Model Zoo, URL: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md 2019.
[39] Tensorflow, Tensorflow Object Detection Api, URL: https://github.com/tensorflow/models/tree/master/research/object_detection 2019.
[40] Tensorpack, Github: Tensorpack faster r-Cnn, URL: https://github.com/tensorpack/tensorpack/tree/master/examples/FasterRCNN 2019.
[41] ultralytics, Yolo v5 on Github, URL: https://github.com/ultralytics/yolov5 2020.
[42] D. Wang, C. Li, S. Wen, Q.L. Han, S. Nepal, X. Zhang, Y. Xiang, Daedalus: breaking non-maximum suppression in object detection via adversarial examples, arXiv (2020) preprint arXiv:1902.02067.
[43] Waymo, Waymo Open Dataset Challenge, URL: https://waymo.com/open/challenges 2020.
[44] W. Zhang, Lyft 3d Object Detection for Autonomous Vehicles, URL: https://www.kaggle.com/c/3d-object-detection-for-autonomous-vehicles/discussion/122820 2020.
[45] H. Zhou, Z. Li, C. Ning, J. Tang, Cad: scale invariant framework for real-time object detection, Proceedings of the IEEE International Conference on Computer Vision 2017, pp. 760–768.