IdentifyNet For Non-maximum Suppression

SHENWANG JIANG¹, TINGFA XU¹, JIANAN LI¹, BO HUANG¹, JIE GUO¹² AND ZIYANG BIAN¹

¹Beijing Institute of Technology, 5 South Zhongguancun Street, Haidian District, Beijing, CO 100081, China. (e-mail: 3120160304@bit.edu.cn)
²Nanjing Research Institute of Electronic Technology, Gulou District, Nanjing, CO 210013, China

This work was supported by the Major Science Instrument Program of the National Natural Science Foundation of China under Grant 61527802, and the General Program of National Nature Science Foundation of China under Grants 61371132 and 61471043.

ABSTRACT Two-stage object detectors have achieved great success in recent years. However, recent work mostly focuses on optimizing loss functions or learning multi-level feature representation, while introducing additional homogeneous task to improve detection has been under-explored. In this paper, a novel framework named as IdentifyNet is proposed, which incorporates an additional identification task to enhance the feature learning of region proposals. Specifically, besides classification and bounding box regression, the proposed IdentifyNet further learns to predict whether two different region proposals belong to the same object, thus forcing the network to learn more informative and representative features for different proposals, especially for those from the same object class. Moreover, current detectors apply greedy non-maximum suppression to remove duplicated boxes whenever their Interaction-over-Union (IoU) exceeds a preset threshold, which would fail when two boxes largely overlap with each other while belonging to two different objects of the same class. To overcome this, we further propose a novel decode non-maximum suppression algorithm by taking advantage of the predicted identity information of different proposals from the identification task. Extensive experiments on PASCAL VOC 2007, VOC 2012 well demonstrate the proposed method can greatly improve detection performance.

INDEX TERMS object detection, non-maximum suppression, deep learning

I. INTRODUCTION

OBJECT detection is at the core of computer vision, which aims at locating objects of certain categories in the images. Recently, deep learning based object detection [1]–[8] has made rapid progress in terms of speed and accuracy thanks to several useful modules. For instance, rapid revolutions of convolutional networks such as VGG, ResNet, ResNeXt and DensNet [9]–[14] have improved detector's ability to extract valid features from input images; RPN [2] helps to propose potential regions of high quality; FPN [15] helps detectors adapt to multi-scale objects. The key improvements of object detection are to introduce additional tasks [2], [3] for realizing and improving traditional algorithms using convolutional networks. Such being the case, is there still be traditional algorithms in object detection can be replaced by convolutional networks without additional annotations? And how can we convert the traditional algorithms to a task of convolutional networks?

Compared with other heated various modules [1], [2], [16], [17] of the two-stage object detection, the non-maximum suppression (NMS) that suppresses the boxes from the same object attracts less attention. It is the last traditional artificial algorithm that has not been replaced by neural networks yet. Tyrolean network (Tnet) [18] and GossipNet (Gnet) [19] have implemented the function of greedy NMS based on ConvNet, but they are still independent modules in object detectors the same as greedy NMS. The principle of traditional NMS is to eliminate repeated boxes through the spatial relations of different boxes. It is inhomogeneous with object detection, which locates objects of certain categories based on the feature map of images. In this paper, the principle of traditional greedy NMS algorithm is changed, thus bringing the additional task of identification. We present a new method for replacing the last traditional artificial algorithm greedy NMS by introducing decode NMS based on the task of identification, and we fully implement the end-to-end two-stage object detector based on deep learning, which is called IdentifyNet.
Prior to IdentifyNet, the introduction of object detection and NMS will be given first. As is mentioned above, object detection is to locate objects of certain categories in the image. In fact, the location of object is an ambiguous question. According to ground-truth bounding box annotations of target objects, the box region of minimum size in the image presenting the body of an object is its location. However, when moving or zooming ground-truth bounding box a little bit, we should also think that there is an object. In the training stage, object detection handles the problem by a given threshold. If the IoU between the detected bounding box and the ground-truth box exceeds the given threshold, the detection results would be considered as containing objects. By this definition, object detectors generate a set of boxes for every object in the test stage, and eliminate repeated boxes of the same object with the help of greedy NMS. The theory of greedy NMS is that high-score box of a specific class eliminates low-score box of the same class, when the IoU between them exceeds a given threshold.

Many improvements [18]–[21] related to NMS have been proposed, but all of them are based on the principle of greedy NMS. In fact, examples can be found from datasets that whether a small part of the body can be annotated as an object depends on whether the whole body is presented. For instance, some typical pictures from PASCAL VOC [22] are presented in Fig. 1, in which the green boxes represent the annotations of the dog. In Fig. 1, the dogs in the first row are occluded, only the head is visible and thus be annotated. However, ground-truth annotations (green boxes) in the second row cover the whole body of the dog, whose head (red boxes) is no longer annotated separately. The variance of object’s annotations causes the boxes generated by object detection for the same object are not very close in space. In other examples shown in Fig. 2, when the IoU between two objects exceeds the threshold of NMS, the greedy NMS will eliminate the boxes more suitable for other objects and leave an edged box to represent the object, or directly eliminate all the boxes of the object. Methods based on greedy NMS are not able to solve these problems.

The purpose of NMS is to eliminate the repeated boxes of one object. Those repeated boxes have a common characteristic, that they are from the same object. That characteristic can not be represented by greedy NMS. We introduce IdentifyNet for NMS by identifying whether the boxes are from the same object and by keeping the highest-score box of the same object. Detected boxes are embedded into box pairs, and correlation and bin correlation feature fusion methods are introduced into IdentifyNet. A small branch network is introduced into two-stage detector to distinguish whether the two boxes of the box pair are from the same object. Judging whether the boxes are from the same object or not is more difficult than classifying the boxes into different classes. It helps the base network extract more special feature maps from the image and the feature maps can also improve the performance of object detection. The number of box pairs is a huge amount, which is approximately equal to the square of the number of detected boxes. Processing the huge amount of box pairs is time and GPU memory consuming. For reducing it, we eliminate the box pair when two boxes in that box pair are not adjacent to each other. We evaluate the proposed method on VOC2007 and VOC2012 [22]. The results show that the proposed method can notably improve the detection performance compared with the baseline model, which demonstrates the effectiveness of our method.

II. RELATE WORK
A. TWO-STAGE OBJECT DETECTION
As the earliest two-stage object detector, R-CNN’s [23] speed is not entirely satisfactory. It processes the potential regions one by one, which brings heavy computation. The fast R-CNN [1] and SPP-net [24] advance ROI pooling and spatial pyramid pooling through extracting all feature maps of potential regions at one time, thus reducing computation. NoCs [25] discovers that the deep convolutional per-region classifier behind the base CNN is of particular importance for object detection. The faster R-CNN [2] proposes ConvNet-based region proposal to replace the classic region proposal. It trains the two-stage detector end to end and improves the speed and performance of object detection. DeNet [26]
III. METHODS

Our method is composed of box pair, feature fusion, IdentifyNet, loss functions, and decode NMS. To reduce processing time and GPU memory, we lay down rules for constituting valid box pairs. To fuse feature maps of box pair, four feature fusion methods are introduced, namely feature concatenation, feature subtraction, feature correlation and bin feature correlation. The IdentifyNet can save a lot of processing time and GPU memory by changing feature fusion location. For handling the imbalance between the true and false sample of the dataset, we adopt focal loss as our loss function, and we have two forms of loss functions. Moreover, decode NMS is introduced for eliminating repeated boxes.

A. BOX PAIR

In the test stage, the RPN generates \( M \) \( BOXES \subset \mathbb{R}^4 \). If no rule is made for constituting valid box pairs, we will get \( M^2 \) box pairs. The consequence would be that much time and GPU memory were wasted to handle the huge amount of box pairs. We draw on the rule of greedy NMS, and only when the IoU (Intersection over Union) between the boxes exceeds a given threshold \( T_{os} \) will the box pair be processed. It can be computed by

\[
box_{pair} = \begin{cases} 
1, & IoU(A,B) > T_{os} \\
0, & \text{otherwise}
\end{cases}
\]  

where \( A, B \in BOXES \subset \mathbb{R}^4 \). When the \( box_{pair} \) is 1, the box pair is processed; when the \( box_{pair} \) is 0, the box pair is skipped, hence saving plenty of processing time and GPU memory.
In the training stage, the $T_{os}$ is set below 0 for getting more training samples. If box $A \in BOXES \subset R^4$ belongs to the ground truth object $GT \in GTES \subset R^4$, both of them should meet the rule attained by:

$$\begin{align*}
\{ & \text{IoU}(A, GT) > GT_{\text{Thresh}} \\
& \text{IoU}(A, GT) = \max(\text{IoU}(A, GTES)) \}
\end{align*} \quad (2)$$

where according to faster R-CNN, we set the $GT_{\text{Thresh}}$ to 0.5.

**B. FEATURE FUSION**

How to compare feature maps of box pair is crucial to judging whether they are same or not. We introduce four methods for fusing the feature maps of box pair. For instance, we have a box pair, which contains two boxes $A$ and $B$. $f_A$ and $f_B$ represent the feature map of $A$ and $B$ respectively. The feature maps of box are extracted by RoI pooling or RoI align. They have the same dimension, which is set as $N \times N \times C$. In this case, $N$ stands for space dimension, and $C$ stands for channel’s number of $f_A$ and $f_B$.

1) Feature Subtraction

The feature subtraction is a form of similarity measurement, as formulated in

$$f_{\text{fuse}} = f_A - f_B \quad (3)$$

When feature subtraction is adopted, the shape of $f_{\text{fuse}}$ is $N \times N \times C$, which is the same as $f_A$. The feature fusing method is to compare the margin of two feature maps.

2) Feature Concatenation

The feature concatenation can be formulated in

$$f_{\text{fuse}} = \text{concat}(f_A, f_B) \quad (4)$$

When feature concatenation is adopted, the shape of $f_{\text{fuse}}$ is $N \times N \times (2 \times C)$, which is the double of $f_A$. The fusion method concatenates $f_A$ with $f_B$.

3) Feature Correlation

Feature maps from the same object have strong correlation. The feature maps of boxes have always been mapped into $N \times N$ grid by RoI pooling or RoI align. We can get the relations among corresponding grids.

$$f_{\text{fuse}}^{i,j,c} = \text{corr}(f_A^{i,j}, f_B^{i+c\%N-N/2,j+c\%N-N/2}) \quad (5)$$

where $i$, $j$ and $c$ represent the $i$th row, $j$th column and $c$th channel of feature map $f_{\text{fuse}}$ respectively. The channel of $f_{\text{fuse}}$ is $N \times N$.

When feature correlation is adopted, the shape of $f_{\text{fuse}}$ is $N \times N \times (N \times N)$, and $N \times N \ll C$.

4) Bin Feature Correlation

One problem of feature correlation is that feature $f_B$ needs to be padded when $i$, $j$ is not in the center of $f_{\text{fuse}}$. However, the padded feature, shown in Fig. 4a, has no important information compared with the original feature. As a result, the correlation feature map has many useless information. To address it, we propose bin feature correlation, which is presented in Fig. 4b. Every position of feature map $f_A$ and $f_B$ is regarded as a bin. Bin feature correlation computes the correlation bin by bin, which is formulated in

$$f_{\text{fuse}}^{i,j,c} = \text{corr}(f_A^{i,j}, f_B^{c\%N,c\%N}) \quad (6)$$

where $i$, $j$ and $c$ represent the $i$th row, $j$th column and $c$th channel of feature map $f_{\text{fuse}}$ respectively. The shape of $f_{\text{fuse}}$ when using bin feature correlation is the same as the shape with feature correlation.

After getting $f_{\text{fuse}}$, we get the similarity score by a ConvNet

$$S_{AB} = \text{ConvNet}(f_{\text{fuse}}) \quad (7)$$

**C. IDENTIFYNET**

Faster R-CNN is chosen as our detection model and FPN based on ResNet 101 as basis net. RPN of faster RCNN generates 300 RoIs in the test stage and 2000 RoIs in training stage. The scale of RPN’s anchor is set at (8, 16, 32) and the ratio of the anchor at (0.5, 1, 2). Feature maps of RoI are extracted from multi-scale feature maps of the image by adopting RoI align and concatenated as final feature map of RoI. The RoI align is set to 7*7 size. We insert a fully convolution layer and a fully connected layer after the RoI align as Identify Net, which is the same as classify net of Faster R-CNN. In the training stage, 128 RoIs are chosen from 2000 RoIs to train the classification net and regression net. Boxes with IoU between the boxes and ground truth boxes exceeding threshold $T_o$ are selected from the 128 RoIs to train IdentifyNet. In the test stage without ground truth, the 300 RoIs generated by RPN will constitute 90000 box pairs. IdentifyNet processes those box pairs would cost much time and GPU memory. Although some pairs are removed according to the rules introduced in Box pair, the remaining box pairs are still large in number. The fully convolutional network processes feature maps of the 300 RoIs to get the feature map of every box. We fuse the feature maps of every remaining box pair, and process them with a fully connected layer to get the score of similarity $d_{ij}$, which $d_{ij}$ represents the similarity score between the $i$th and the $j$th RoIs. For eliminated pairs, the similarity is set to 0.

**D. LOSS FUNCTION**

When the box meets the Eq. 2, we consider it is from ground truth GT. According to the definition, many sets of boxes $GT_{i,j}(i \leq N, j \leq M)$ can be obtained. Here $N$ represents the number of classes in the dataset; $M$ represents...
FIGURE 4: The comparison of Correlation fusion methods and Bin Correlation fusion methods. The red dotted box represents the position of generated feature maps, and the purple dotted boxes represent the feature maps to be fused.

FIGURE 5: The comparison of pairwise embedding in different positions. \(N\) represents the number of boxes’ feature maps; \(M\) represents the number of box pairs’ feature maps, and \(M \gg N\). The green arrow represents light computation, and the red arrow represents heavy computation. The first row represents the pairwise embedding before the ConvNet, and the second row changes the position.

the number of objects in images; \(i\) represents the \(i\)th class and \(j\) represents the \(j\)th object.

The loss function has two forms. One identifies the boxes from all the other boxes, and the other identifies them merely from boxes whose objects are from the same class.

Based on the first form, the label of box pair can be put as follows:

\[
label = \begin{cases} 
  i & A, B \in GT_{ij} \\
  0 & \text{otherwise}
\end{cases}
\]

(8)

where \(A\) represents one box of a box pair, and \(B\) represents the other box of the box pair. The loss of identify can be computed as:

\[
Loss_{id} = L(\text{sigmoid}(y), label)
\]

(9)

Based on the second form, we can get the mask of the box pair whose boxes come from the same class, which is put as follows:

\[
mask = \begin{cases} 
  1 & A, B \in GT_{i*} \\
  0 & \text{otherwise}
\end{cases}
\]

(10)

the loss of identify can be computed as:

\[
Loss_{id} = L(\text{sigmoid}(y(mask)), label(mask))
\]

(11)
Since false sample is rare in the dataset for training IdentifyNet, focal loss [32], which is able to effectively handle sample imbalance, is adopted as loss function.

The final loss is composed of detection loss \( \text{loss}_{\text{det}} \) and identify loss \( \lambda \text{loss}_{\text{id}} \), as is formulated in

\[
\text{loss} = \text{loss}_{\text{det}} + \lambda \text{loss}_{\text{id}} \tag{12}
\]

### Algorithm 1 Decode NMS

1. **Input**
   - \( B = \{b_1, b_2, \ldots, b_N\} \), the detection boxes
   - \( S = \{s_1, s_2, \ldots, s_N\} \), the scores of \( B \)
   - \( D = \{d_1, d_2, \ldots, d_N\} \), the similarity scores of \( B \)
   - \( T \), the relationship threshold
   - \( N \), the number of boxes

2. **Output**
   - \( K \), the remaining boxes

3. **Begin**
4. \( K \leftarrow \{\} \)
5. \( NK \leftarrow \text{Zeros}(N) \)
6. \( \text{Order} \leftarrow \text{sort}(S) \)
7. **for** \( i \) in \( \text{Order} \) **do**
8.  **if** \( \text{NK}[i] == 0 \) **then**
9.    \( K \leftarrow i \)
10.    \( NK \leftarrow NK + D[i, :] \)
11.  **end if**
12. **end for**
13. **End**

### E. DECODE NMS

Faster R-CNN merely generates the detection boxes \( B \) and the corresponding class scores \( S \). The greedy NMS eliminates extra boxes based on the spatial relationship between boxes. It selects the box with high-score and eliminates other boxes when the IoU between them exceeds a given threshold \( T_N \). Essentially, \( T_N \) is equal to \( T_{os} \) in Eq. 1, but \( T_{os} \) stresses on reducing the computation. We eliminate extra boxes by the similarity score \( D \) and \( S \). The Decode NMS selects high-score box \( b_{h} \), and eliminates box \( b_{l} \) when the similarity score of them exceeds a given threshold \( T_S \). The pseudo-code of decode NMS is presented in Algorithm 1.

### IV. EXPERIMENTS

All the experiments are conducted on PyTorch [33] and the models are trained on 3 * GTX1080TI GPU, E5-2620V4 CPU and 32G DDR4. Using stochastic gradient descent, we optimize the IdentifyNet, and set the momentum to 0.9, the learning rate to 0.001, the batch size to 6, and the epochs of training to 10. All of our experiments are carried out in the above settings.

### A. PERFORMANCE COMPARISON

Table 1 shows the results of the IdentifyNet with different NMS methods and the baseline on PASCAL VOC 2007 test dataset. Table 2 shows the results on PASCAL VOC 2012 test dataset. The model is trained on PASCAL VOC2007 and 2012 trainval dataset. According to the table, the IdentifyNet with decode NMS achieves 86.9% mAP, and the IdentifyNet with greedy NMS achieves 86.8% mAP. Since the baseline only achieves 84.7% mAP, it is proved that our method can significantly improve the performance. In this feature fusion method, the performance of IdentifyNet with greedy NMS is slightly better than the decode NMS. However, the decode NMS outperforms greedy NMS in ABLATION STUDY.

### B. ABLATION STUDY

We comprehensively verify our method in different parameter settings on the PASCAL VOC 2007 [22] detection benchmark.

1) Feature Fusions

We evaluate IdentifyNet with different feature fusion methods, and the results are presented in Table. 3. The bin correlation feature fusion gives the best performance, and it obtains 86.9% mAP with greedy NMS and 86.8% mAP with decode NMS. The IdentifyNet outperforms the baseline. It proves that introducing difficult tasks to detector can improve the performance of detection.

2) Loss functions

We have two forms of loss function. One merely compares the boxes from the same class, which is presented in Eq. 11. The other compares the boxes from all classes presented in Eq. 9. We name the first one ‘mask on’, and the other ‘mask off’. The result is presented in Table. 3 and Fig. 6, and it reveals that ‘mask on’ gives better performance in all fusion methods except correlation fusion method.

3) NMS methods

Table. 3 shows the results of IdentifyNet with greedy NMS and decode NMS, which reveals that decode NMS outperforms greedy NMS in almost all conditions.

Fig. 7 present the results of IdentifyNet with greedy NMS and decode NMS on different IoU threshold \( T_{os} \). Compared with decode NMS, the results of greedy NMS are closely related to the threshold \( T_{os} \). The decode NMS with feature correlation fusion method can work on almost all IoU thresholds, but the SUB, CC fusion methods are sensitive to the threshold \( T_{os} \). All the solid lines are above the dotted lines, which proves our method can efficiently eliminate duplicate objects.

Fig. 8 shows the mAP@[0.5:0.95] of IdentifyNet with greedy NMS and decode NMS. From it, we can know that decode NMS outperforms the greedy NMS.
The pairwise embedding position has a major impact on detection speed. Since the concatenation and subtraction fusion methods cannot decrease the size of fusion feature maps, the two fusion methods will make IdentifyNet beyond the memory of the GTX1080TI. By the correlation fusion method, we compare different pairwise embedding positions. The results are presented in Fig. 9, which shows the detection speed is much more rapid when the fusion feature is computed after ConvNet. When the fusion feature is computed before ConvNet, detection speed is closely related to the IoU threshold $T_{os}$. Specifically, ConvNet needs to process a large amount of fusion feature maps when the fusion feature is computed after ConvNet, and the number of fusion feature maps is closely related to $T_{os}$. When the fusion feature is computed after ConvNet, ConvNet needs to fuse only a few feature maps, so the detection speed is weakly related to $T_{os}$. When the fusion method is `correlation + concatenation’, the detection speed is related to $T_{os}$. The reason lies in the fact that combining the two methods will increase computation, and lifting the IoU threshold will decrease the amount of fusion feature maps needed to be processed.

4) Pairwise embedding positions

The pairwise embedding position has a major impact on detection speed. Since the concatenation and subtraction fusion methods cannot decrease the size of fusion feature maps, the two fusion methods will make IdentifyNet beyond the memory of the GTX1080TI. By the correlation fusion method, we compare different pairwise embedding positions. The results are presented in Fig. 9, which shows the detection speed is much more rapid when the fusion feature is computed after ConvNet. When the fusion feature is computed before ConvNet, detection speed is closely related to the IoU threshold $T_{os}$. Specifically, ConvNet needs to process a large amount of fusion feature maps when the fusion feature is computed after ConvNet, and the number of fusion feature maps is closely related to $T_{os}$. When the fusion feature is computed after ConvNet, ConvNet needs to fuse only a few feature maps, so the detection speed is weakly related to $T_{os}$. When the fusion method is `correlation + concatenation’, the detection speed is related to $T_{os}$. The reason lies in the fact that combining the two methods will increase computation, and lifting the IoU threshold will decrease the amount of fusion feature maps needed to be processed.

FIGURE 6: The comparison of ‘mask on’ loss function and ‘mask off’ loss function with different fusion methods on IdentifyNet. The evaluation standard is mAP@[0.5:0.95]. SUB, CC, CR and BCR represent the feature subtraction, feature concatenation, feature correlation and bin correlation respectively.

V. CONCLUSION

In this paper, hard task is introduced to enhance the feature of objects and the object detection performance. The hard task must be homogeneous with object detection, so we choose identifying each object as the hard task, which, however, will increase the computation in object detection. To avoid that, we introduce the Correlation and Bin Correlation to fuse feature maps. Changing the location of fusing feature maps can also reduce computation. With the hard task, the last artificial algorithm greedy NMS can be replaced by our decode NMS.

In this paper, FPN is chosen as our baseline, and ResNet101 as the base network. Detailed experiments on PASCALVOC are conducted to prove that our method can significantly improve detector’s performance.
FIGURE 7: The comparison of greedy NMS and decode NMS with different fusion methods on IdentifyNet and the loss function is ‘mask on’. The dotted lines represent the greedy NMS and the solid lines represent the decode NMS. Different colors represent different fusion methods. ‘FPN+G’ represents the baseline. SUB, CC, CR and BCR represent the feature subtraction, feature concatenation, feature correlation and bin correlation respectively.

ACKNOWLEDGMENT
Shenwang Jiang would like to thank Tingfa Xu, Bo Huang, Jianan Li, Jie Guo and Ziyang Bian for the support. The authors wish to thank the anonymous reviewers for their valuable suggestions.

REFERENCES


ZIYANG BIAN received the M.S degree in School of Optics and Photonics from Beijing Institute of Technology, Beijing, China, in 2016. He is currently pursuing the PH.D degree in School of Optics and Photonics from Beijing Institute of Technology, Beijing, China. His research interests include object detection and related computer vision problems.

***