PSRR-MaxpoolNMS: Pyramid Shifted MaxpoolNMS with Relationship Recovery

Tianyi Zhang[1] Jie Lin[1*] Peng Hu[2] Bin Zhao[3] Mohamed M. Sabry Aly[4]
[1] I2R, A*star, Singapore [2] Sichuan University, China [3] IME, A*star, Singapore [4] NTU, Singapore
{zhang_tianyi, lin-j}@i2r.a-star.edu.sg, penghu.ml@gmail.com, zhaobin@ime.a-star.edu.sg, msabry@ntu.edu.sg

Abstract

Non-maximum Suppression (NMS) is an essential post-processing step in modern convolutional neural networks for object detection. Unlike convolutions which are inherently parallel, the de-facto standard for NMS, namely GreedyNMS, cannot be easily parallelized and thus could be the performance bottleneck in convolutional object detection pipelines. MaxpoolNMS is introduced as a parallelizable alternative to GreedyNMS, which in turn enables faster speed than GreedyNMS at comparable accuracy. However, MaxpoolNMS is only capable of replacing the GreedyNMS at the first stage of two-stage detectors like Faster-RCNN. There is a significant drop in accuracy when applying MaxpoolNMS at the final detection stage, due to the fact that MaxpoolNMS fails to approximate GreedyNMS precisely in terms of bounding box selection. In this paper, we propose a general, parallelizable and configurable approach PSRR-MaxpoolNMS, to completely replace GreedyNMS at all stages in all detectors. By introducing a simple Relationship Recovery module and a Pyramid Shifted MaxpoolNMS module, our PSRR-MaxpoolNMS is able to approximate GreedyNMS more precisely than MaxpoolNMS. Comprehensive experiments show that our approach outperforms MaxpoolNMS by a large margin, and it is proven faster than GreedyNMS with comparable accuracy. For the first time, PSRR-MaxpoolNMS provides a fully parallelizable solution for customized hardware design, which can be reused for accelerating NMS everywhere.

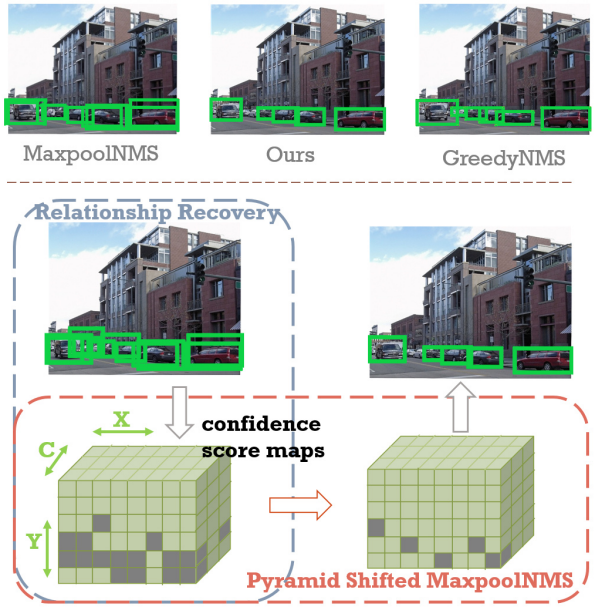

Figure 1. (Top) Visualized comparison of MaxpoolNMS [2], our method and GreedyNMS at the final detection stage of Faster-RCNN. Compared to MaxpoolNMS, our method behaves more like GreedyNMS. (Bottom) Pipeline of PSRR-MaxpoolNMS. Relationship Recovery to build up the confidence score maps, followed by Pyramid Shifted MaxpoolNMS to eliminate overlapped boxes and only keep the boxes with peak scores. Each cell on the map encodes the confidence score, scale/ratio (C) and spatial location (X, Y) of bounding boxes.

1. Introduction

Object detection is one of the key tasks in computer vision, with the objective of localizing and classifying objects in a scene. During the past few years, deep convolutional neural networks has emerged as the champion of object detection [9, 21, 19]. Convolutional object detectors are broadly grouped into either one-stage detectors like SSD [19] and YOLO [20] or two-stage detectors like Faster RCNN [21] and R-FCN [3], in which convolutions often account for the majority of computing operations. On the other side, significant progress has been made towards better-performing dedicated hardware for accelerating the convolution operations by exploiting their inherent parallelism, such as GPU and Google TPU [15]. Therefore, the execution time spent on convolution operations is decreas-

*Corresponding author: J. Lin (lin-j@i2r.a-star.edu.sg)
This research is supported by the Agency for Science, Technology and Research (A*STAR) under its AME Programmatic Funds (Project No.A1892b0026).

ing rapidly, e.g., at milliseconds.

Non-maximum Suppression (NMS), as a must-have post-processing technique in all convolutional object detectors, is likely to become the performance bottleneck in object detection pipelines [2]. The de-facto standard for NMS, namely GreedyNMS, is composed of a sorting operation over confidence scores for tens of thousands of bounding boxes, followed by nested for loops to greedily select the boxes with high scores and remove the boxes significantly overlapped with the selected ones. Unlike convolutions which are inherently parallel, GreedyNMS cannot be easily parallelized due to the nested for loops. Thus, GreedNMS would gradually dominate the execution time of convolutional object detectors [2], as convolutions run faster thanks to the increasing parallelism on dedicated hardware (e.g., from P100 to V100 GPU).

MaxpoolNMS [2], as a parallelizable alternative to GreedyNMS, is introduced to largely accelerate GreedyNMS without incurring loss in detection accuracy. MaxpoolNMS is inspired by the observation that bounding boxes with high confidence scores correlate to peak values on the so-called confidence score maps in which the spatial relationship between anchor boxes is preserved. Therefore, NMS can be designed as simple max pooling on the score maps which encode confidence scores, scales, ratios, and spatial locations of anchor boxes (see Fig 1 Bottom). After max pooling, only boxes with peak scores are kept and the others are suppressed. In terms of execution time, MaxpoolNMS runs much faster than GreedyNMS, mainly attributed to the fact that max pooling operations are inherently parallel.

However, MaxpoolNMS is dedicated only to replacing the GreedyNMS at the first stage of two-stage detectors, e.g., the GreedyNMS after the region proposal network in Faster-RCNN. There is a significant drop in detection accuracy when directly applying MaxpoolNMS at the second stage of two-stage detectors, e.g., the GreedyNMS after the detection network in Faster-RCNN (see visualized example in Fig. 1 Top and quantitative results in Table 1). This lowers the value of MaxpoolNMS since a customized hardware for MaxpoolNMS cannot be reused to replace GreedyNMS at all stages in all detectors.

In this paper, we propose a general approach, namely PSRR-MaxpoolNMS, to completely replace GreedyNMS at all stages in all detectors. The key is to approximate GreedyNMS as precisely as possible, which can be measured by the overlap ratio of the selected bounding boxes between GreedyNMS and the approximation method. As evidenced by the low overlap ratio (see Table 1), MaxpoolNMS fails to approximate GreedyNMS, mainly due to (A) the score map mismatch problem on confidence score maps (see Fig. 2) and (B) the difficulty of maximizing the score map sparsity (see Fig. 4 and Fig. 5) with a single-scan max pooling on the confidence score maps. PSRR-MaxpoolNMS introduces a Relationship Recovery module and a Pyramid Shifted MaxpoolNMS module to solve problem (A) and (B) respectively. As a result, our PSRR-MaxpoolNMS is able to approximate GreedyNMS more precisely than MaxpoolNMS (see the overlap ratio in Table 1).

We summarize our contributions as follows:

- A general approach PSRR-MaxpoolNMS to accelerate NMS at all stages in all convolutional object detectors.
- A Relationship Recovery module to correct the score map mismatch when projecting bounding boxes to confidence score maps, enabling more accurate scale, aspect ratio and spatial relationships between boxes.
- A Pyramid Shifted MaxpoolNMS on confidence score maps to significantly increase the sparsity of the score maps, and thus eliminate more overlapped boxes.
- In PSRR-MaxpoolNMS, the Relationship Recovery and the Pyramid Shifted MaxpoolNMS are simple and parallelizable operations. Therefore, for the first time, PSRR-MaxpoolNMS provides a fully parallelizable solution for customized hardware design, which can be reused for accelerating NMS everywhere.
- Finally, our PSRR-MaxpoolNMS outperforms MaxpoolNMS by a large margin. Moreover, it is proven faster than GreedyNMS with comparable accuracy.

2. Related Works

2.1. One-stage and Two-stage Object Detectors

Convolutional object detection frameworks are roughly classified into One-stage detectors and Two-stage detectors. One-stage detectors like SSD and YOLO [19, 20, 17] directly predict the bounding box coordinates and class probability by passing an entire image through a single unified network. Two-stage detectors like Faster-RCNN [21, 11, 9] are based on the class agnostic region proposals. The region proposals are the candidate bounding boxes that potentially enclose target objects. Different from previous works RCNN [10] or Fast-RCNN [9] which employ hand-crafted region proposal generation [25], Faster-RCNN [21] generates region proposals by training a Region Proposal Network (RPN). The features of the proposals are fed into subsequent detection network to predict the final box coordinates and class-specific probability for each proposal.

2.2. Non-maximum Suppression

The final goal of object detectors is to output exactly one bounding box to tightly enclose each target object. However, most object detection pipelines tend to generate redundant highly-overlapped bounding boxes to enclose an object, hence introducing large number of false positives.

Non-maximum Suppression (NMS) is an essential step to suppress the redundant bounding boxes.

The most widely used NMS method is GreedyNMS [4]. GreedyNMS firstly sorts the boxes by their confidence scores in descending order, then iteratively selects the most confident predictions from the remaining boxes and eliminates all the other boxes that have large overlap with the selected ones. There are variants of NMS to increase the detection accuracy [1, 18, 13, 7, 22]. SoftNMS [1] decreases the scores of the boxes to be suppressed, instead of deleting these boxes by hard thresholding. Adaptive NMS [18] learns to adaptively set the box selection threshold according to object density. Hosang *et al.* [13] reformulates NMS as ConvNet that can be trained end to end. Visibility guided NMS [7] leverages the detection of the whole objects as well as the detection of the visible parts to tackle the problem of highly occluded object detection. FeatureNMS [22] leverages on the feature embedding distance to determine whether to suppress or keep the candidate boxes.

Hardware-aware NMS acceleration has been less explored. MaxpoolNMS [2] reformulates NMS as max pooling on confidence score maps to remove the redundant boxes. Max pooling operations are inherently parallel, thus MaxpoolNMS is much more efficient than GreedyNMS which cannot be easily parallelized. However, MaxpoolNMS is only confined to the region proposal network (RPN) of the two-stage detectors, and cannot be generalized to all stages in all detectors including one-stage detectors.

3. Method

In this section, we first briefly review MaxpoolNMS [2] (Section 3.1) and analyze its limitations (Section 3.2). Then we introduce our PSRR-MaxpoolNMS to address the limitations. PSRR-MaxpoolNMS is composed of two steps: Relationship Recovery (Section 3.3), followed by Pyramid Shifted MaxpoolNMS (Section 3.4).

3.1. Revisiting MaxpoolNMS

MaxpoolNMS [2] is an effective yet efficient NMS approach which is specifically designed for removing the overlapped anchor boxes at the first stage of Faster-RCNN detection pipeline, i.e., the Region-Proposal Network (RPN). MaxpoolNMS is composed of two modules. First, as illustrated in Fig. 2, it constructs a set of confidence score maps, of which each score map corresponds to a specific combination of anchor box scale and ratio (i.e., channel c), and each cell on the score map encodes the objectness score (i.e., cell value) and spatial location (i.e. x and y on the map) of an anchor box that generated by the RPN. For instance, if we use 4 anchor box scales $\{64^2, 128^2, 256^2, 512^2\}$ and 3 anchor box ratios $\{1:2, 1:1, 2:1\}$ for a RPN with down sampling ratio β (e.g., $\beta = 16$), there are 12 confidence score maps with width

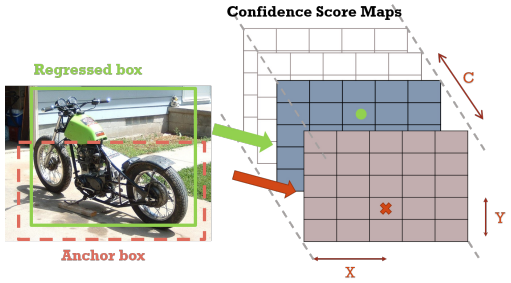

Figure 2. MaxpoolNMS [2] projects a anchor box (Dashed red) to a score map with ratio 1:2 (Coral), without consideration of box regression. This leads to the **score map mismatch** problem, i.e. the regressed box (Solid green) of the anchor has changed in ratio, and thus is projected to another score map with ratio 1:1 (Blue gray). The projection of the regressed box is correct as it encloses the motorcycle more accurately than its corresponding anchor box.

$\lfloor \frac{W}{\beta} \rceil$ and height $\lfloor \frac{H}{\beta} \rceil$, where W and H denote image width and height respectively.

Second, based on the observation that objects correspond to peak scores on the confidence score maps, a simple max pooling is operated on the maps to suppress anchor boxes with low scores and only keep anchor boxes with peak scores. Moreover, since each score map is dedicated to a specific anchor box size, the kernel size and pool stride for max pooling on that map are determined by its associated anchor box size,

$$k_x, s_x = \max(\lfloor \frac{\alpha w}{\beta} \rceil, 1), \quad k_y, s_y = \max(\lfloor \frac{\alpha h}{\beta} \rceil, 1) \quad (1)$$

where k_x,k_y are the kernal sizes and s_x,s_y are the pool strides in x direction and y direction respectively. w,h denote the anchor box size (width and height) on a specific score map. β is the down sampling ratio for the score maps. α represents the overlap threshold, which is used to control the trade-off between precision and recall. A larger α would suppress more overlapped boxes (lead to higher precision), but at the risk of missed object detections (lead to lower recall).

Moreover, the max pooling in MaxpoolNMS has 3 variants: (1) **Single-Channel MaxpoolNMS**, or multi-scale MaxpoolNMS, applies max pooling on each score map (channel) independently. (2) **Cross-Ratio MaxpoolNMS** concatenates score maps at different ratios for each scale, followed by 3D max pooling on the concatenated maps. (3) **Cross-Scale MaxpoolNMS** concatenates score maps at adjacent scales for each ratio, followed by 3D max pooling on the concatenated maps.

Finally, the anchor boxes remaining on the score maps are combined and sorted by their scores in descending order. Only the top boxes are returned as final detections.

3.2. Limitations of MaxpoolNMS

Though the execution of MaxpoolNMS is much faster than GreedyNMS by paralleling the simple max pooling operations, MaxpoolNMS suffers from a huge shortcoming that it is dedicated only to replacing GreedyNMS at the first stage of regular two-stage convolutional object detectors. To maintain high detection accuracy, GreedyNMS is still a must-have post processing method for the second stage of two-stage detectors and one-stage detectors such as SSD. This makes MaxpoolNMS less attractive in the sense that it lowers the value of a customized hardware for MaxpoolNMS which cannot be easily reused for accelerating NMS at all stages in all detectors.

We observe the detection accuracy drops significantly when applying MaxpoolNMS at the second stage of two-stage detectors. Specifically, we perform MaxpoolNMS after the detection network of Faster-RCNN to remove overlapped boxes, with ResNet-50 as backbone. As shown in Table 1, MaxpoolNMS performs significantly worse than GreedyNMS on PASCAL VOC dataset, with over 50% drop in mAP. As shown in Fig. 1 Top, one can see that the final selected boxes of MaxpoolNMS is significantly different from that of GreedyNMS, which leads us to a hypothesis that the poor performance of MaxpoolNMS is because it fails to approximate GreedyNMS very well. We measure the quality of the approximation as the overlap ratio of selected bounding boxes between MaxpoolNMS and GreedyNMS. As evidenced in Table 1, mAP increases with the overlap ratio, but the overlap ratio for MaxpoolNMS is low.

We find that there are 2 key factors that lead to the low overlap ratio for MaxpoolNMS, the score map mismatch on confidence score maps and the difficulty of maximizing the score map sparsity with a single-scan max pooling on the confidence score maps.

- **Score map mismatch** occurs during the construction of confidence score maps. MaxpoolNMS projects anchor boxes to score maps without consideration of box regression. This leads to the score map mismatch problem if the regressed boxes correspond to the anchor boxes have changed dramatically in location, scale or aspect ratio. Fig. 2 shows one example of change in ratio. The mismatch would cause wrong box projections on score maps, which in turn bring in negative effect on the following max pooling operations.

- **Low sparsity on score maps**. Since MaxpoolNMS operates only a single-scan max pooling on the confidence score maps, it is hard to achieve high sparsity on dense score maps, implying a lot of highly-overlapped boxes remain after pooling, as illustrated in the left of Fig. 4 (i.e., max pooling with Single Channel only). Moreover, a single-scan max pooling on the confidence score maps would cause the edge effect. As shown in Fig. 5 Left, the boxes in the adjacent cells are both kept after max pooling, even though one of them is considered as a duplication.

3.3. Relationship Recovery

Instead of projecting anchor boxes to the confidence score maps, our Relationship Recovery projects the regressed boxes to the maps, which solves the score map mismatch problem. With the help of box regression, the regressed boxes in general enclose the objects more accurately than their corresponding anchor boxes, in terms of spatial location, size and shape (i.e., scale and aspect ratio). As such, the confidence score maps projected by the regressed boxes are able to better reflect the actual spatial and channel (a combination of scale and ratio) relationships between objects in a scene. Concretely, the Relationship Recovery module consists of three parts: spatial and channel recovery to identify the spatial location (X, Y) and the channel $(C(s, r))$ of a regressed box should be mapped to, followed by the score assignment which determines the confidence score for each cell in the maps (see Fig. 3).

Spatial Recovery. MaxpoolNMS projects anchor box to wrong spatial location on score map due to dramatic shift of location after box regression. To address this location mismatch problem, given the center position $[x_c, y_c]$ of a regressed box in input image, Spatial Recovery maps it to the spatial index $[X, Y]$ on score map as

$$X = \lfloor \frac{x_c}{\beta} \rfloor, \quad Y = \lfloor \frac{y_c}{\beta} \rfloor, \tag{2}$$

where β is the down sampling ratio of the score maps.

Channel Recovery. MaxpoolNMS projects anchor box to a channel $(C(s_0, r_0))$ of the score maps simply based on the default scale (s_0) and ratio (r_0) of the anchor box. Similarly, the channel projection could be wrong if the corresponding regressed box have changed dramatically in scale and/or ratio. To solve this channel mismatch problem, given a regressed box with size $w^{'}, h^{'}$, Channel Recovery calculates the nearest scale s to $w^{'} \times h^{'}$ and the nearest ratio r to $\frac{h^{'}}{w^{'}}$ based on Euclidean-distance, and choose $C(s, r)$ as the projected channel for the box.

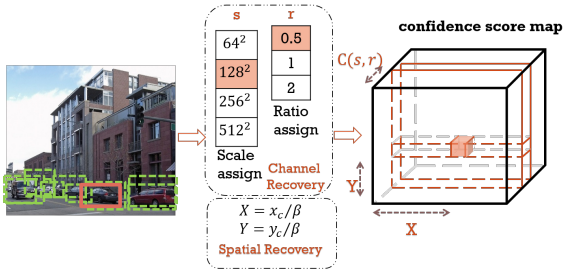

Figure 3. Relationship Recovery to solve score map mismatch.

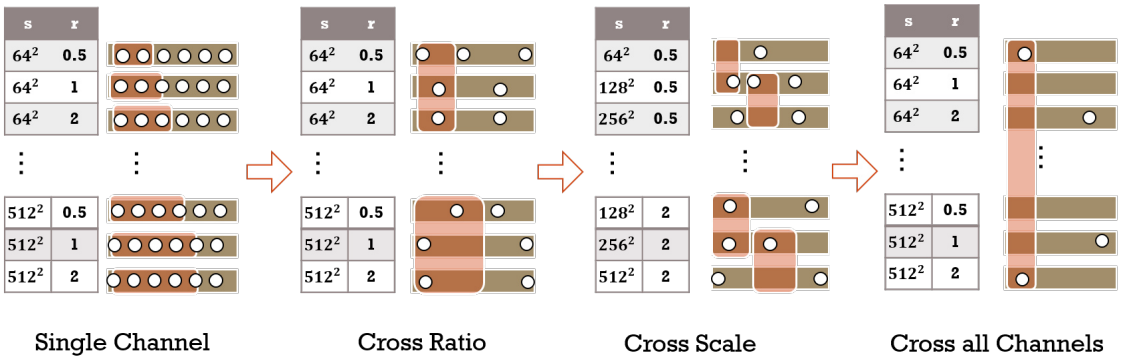

Figure 4. Pyramid MaxpoolNMS. A sequence of max pooling is operated one after another on the confidence score maps, with different channel combinations and pooling parameters (kernel size and stride) determined by the scale (s) and ratio (r) of the map (From left to right: single channel, cross ratio, cross scale and cross all channels). One can see that the confidence score maps become more and more sparse with the Pyramid MaxpoolNMS (i.e., multi-scan max pooling).

Score Assignment. After the spatial locations and channels for all boxes are determined, each cell in the maps could have more than one box projected to it. Therefore, score assignment is introduced to only keep the box with the highest score in each cell. One may note that the score assignment is basically 1×1 max pooling in each cell of the score maps, thus it can be treated as a pre-filtering step for removing overlapped boxes that are easy to be identified.

Remarks. All operations of the relationship recovery method are simple and highly parallelizable. In addition, the relationship recovery method is anchor-free. In other words, relationship recovery, as the first step of our PSRR-MaxpoolNMS, opens up a possibility to extend PSRR-MaxpoolNMS from anchor-based one-stage or two-stage convolutional object detectors to anchor-free convolutional object detectors [16, 5], since the construction of confidence score maps doesn't reply on anchor box at all. Instead, it only requires the location and size of regressed box, which is accessible in anchor-free detectors as well. We would leave it as our future work.

3.4. Pyramid Shifted MaxpoolNMS

We propose Pyramid Shifted MaxpoolNMS to remove overlapped boxes on the confidence score maps, in which the Pyramid MaxpoolNMS aims to thoroughly suppress overlapped boxes across channels (scale and ratio), while the Shifted MaxpoolNMS aims to effectively eliminate overlapped boxes in spatial domain by addressing the edge effect problem. After Pyramid Shifted MaxpoolNMS, the score maps become highly sparse, with only a small number of non-zero cells. The boxes in the non-zero cells are returned as final detections.

Pyramid MaxpoolNMS. On one hand, MaxpoolNMS operates only a single-scan max pooling on the confidence score maps. On the other hand, MaxpoolNMS assumes overlapped boxes only exist in the channels with adjacent scales (or ratios) on the score maps, which is not always true as the overlapped boxes can be distributed at arbitrary scales/ratios (e.g. a mini cooper occluded by a truck). As such, a single-scan max pooling with invalid assumption is not sufficient to suppress overlapped boxes effectively, resulting in low sparsity on the score maps after pooling. One can increase the overlap threshold α in Eq. 1 to induce higher sparsity, at the risk of missed detections [2].

We propose Pyramid MaxpoolNMS to induce score map sparsity progressively by executing a sequence of max pooling one after another on the confidence score maps with different channel combinations, as illustrated in Fig. 4. The sequence of max pooling starts from a Single-Channel max pooling, followed by Cross-Ratio and Cross-Scale max pooling, and ends at Cross-all-Channels max pooling. As introduced in Section 3.1, Single-Channel max pooling operates on single score map independently, while Cross-Ratio and Cross-Scale max pooling operate on multiple score maps by concatenating channels at adjacent ratios/scales. In addition, we introduce Cross-all-Channels max pooling which operates pooling on all channels. In this way, our Pyramid MaxpoolNMS gradually increases the "receptive field" of pooling operator from local (single score map) to global (all maps), thus without the need of any assumption on the distribution of overlapped boxes.

When operating max pooling on single channel independently, the kernel size and stride for each channel are set as Eq. 1. When operating max pooling across multiple channels, the kernel size (or stride) is set as the minimum of kernel sizes (or strides) of the channels concatenated. First, if the kernel size is larger than the minimum value, it may suppress true positives detected by the precedent Single-Channel max pooling. Second, the larger gap between scales/ratios, the less likely to have overlapped boxes, hence a small kernel size (or stride) could reduce the risk of suppressing true positives.

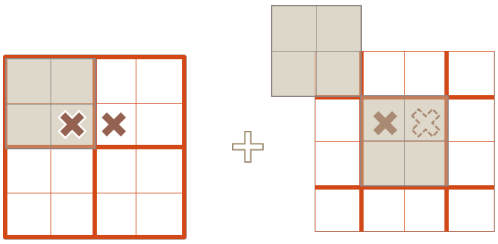

Figure 5. Shifted MaxpoolNMS to alleviate edge effect. (Left) The boxes in the adjacent cells are both kept after max pooling with pool size 2x2 and stride 2. (Right) By adding another max pooling with 1 cell shift, the box with higher score is kept, while the other one is suppressed.

Shifted MaxpoolNMS. Shifted MaxpoolNMS can further increase the score map sparsity, and thus eliminate overlapped boxes in spatial domain (X, Y) more effectively. This is achieved by introducing additional max pooling with a spatial shift on the confidence score maps, which in turn addresses the edge effect problem, as shown in Fig. 5. Specifically, given a kernel size k, the shifted max pooling is operated on the score maps padded with $\lfloor \frac{k}{2} \rceil$ zeros around the border. Finally, the shifted max pooling can be appended after each pooling step in the sequence of Pyramid MaxpoolNMS.

4. Experiments

4.1. Experimental Setup

We evaluate different NMS approaches only at the inference stages of both Faster-RCNN [21] and SSD [19]. (1) Faster-RCNN [21] is a two-stage convolutional object detector. We use ResNet-50, ResNet-101 and ResNet-152 [12] as the backbone network architectures. For Faster-RCNN training, we follow the default training parameters of the public PyTorch implementation [1]. Since MaxpoolNMS [2] can be viewed as a simplified version of our PSRR-MaxpoolNMS, we simply replace GreedyNMS with Multi-Scale (or Single-Channel) MaxpoolNMS as the post processing technique at the first stage of Faster-RCNN, which achieves comparable accuracy but runs much faster than GreedyNMS. (2) SSD [19] is a one-stage convolutional object detector. We use VGG-16 [24], MobileNet-v1 [14], MobileNet-v2 [23] as the backbone. We evaluate NMS using the pre-trained models provided by the public PyTorch implementation [2]. In pre-processing stage, SSD first filters out the bounding boxes with score $<$0.01 for each class. For GreedyNMS, it further selects 200 boxes with top scores from the boxes passed the pre-processing stage. Our PSRR-MaxpoolNMS takes all boxes passed the pre-processing stage as input.

[1] https://github.com/jwyang/faster-rcnn.pytorch
[2] https://github.com/qfgaohao/pytorch-ssd

Table 1. Detection accuracy (mAP) of our method and MaxpoolNMS on Pascal VOC, at the second stage of Faster-RCNN with ResNet-50 as backbone. We also report the overlap ratio of selected bounding boxes between GreedyNMS and MaxpoolNMS (or our method). As a reference, mAP for GreedyNMS is 78.1%.

Method	Box Overlap (%)	mAP (%)
MaxpoolNMS [2] single	15.0	33.0
MaxpoolNMS [2] ratio	18.5	36.6
MaxpoolNMS [2] scale	11.6	26.5
Ours	45.3	77.6

For both Faster-RCNN and SSD, our PSRR-MaxpoolNMS is applied to suppress the final bounding box predictions. We fix α to the value of 0.75. In our channel-recovery step, we set the anchors to the scales of $[64^2, 128^2, 256^2, 512^2]$ and the ratios of $[0.5, 1, 2]$. For Cross-Ratio MaxpoolNMS, all the 3 ratios of each scale are concatenated for max pooling. For Cross-Scale MaxpoolNMS, we only concatenate 2 channels with adjacent scales for each step of max pooling. We denote Single-Channel MaxpoolNMS, Cross-Ratio MaxpoolNMS and Cross-Scale MaxpoolNMS as *single*, *ratio* and *scale*, respectively.

We perform experiments on PASCAL VOC [6] and KITTI [8] datasets. For PASCAL VOC, we train Faster-RCNN detection model using 2007 and 2012 trainval datasets and evaluate on 2007 test dataset. We report the mean average precision (mAP) for PASCAL VOC dataset. For KITTI, we randomly split the dataset into 5611 training images and 1870 testing images. We report mAP at difficulty levels from easy to difficult on KITTI.

4.2. Comparison with MaxpoolNMS

We compare MaxpoolNMS [2] with our PSRR-MaxpoolNMS approach at the second stage of Faster-RCNN. Results on PASCAL VOC dataset are shown in Table 1. First, we observe that MaxpoolNMS performs poorly, e.g. MaxpoolNMS single 33% versus GreedyNMS 78.1%. As expected, though the detection mAP increases with the overlap ratio, the overlap ratio of selected boxes between MaxpoolNMS and GreedyNMS is still very low. Second, our PSRR-MaxpoolNMS better approximates GreedyNMS which is evidenced by the large overlap ratio and comparable detection accuracy with GreedyNMS (less than 1% drop in mAP). It is worth noting that similar to [2], the only parameter to be set for our method is the overlap threshold α. Thus, our method PSRR-MaxpoolNMS does not introduce extra parameter tuning workload, while outperforms MaxpoolNMS by a significant margin.

4.3. Comparisons with GreedyNMS

We compare our PSRR-MaxpoolNMS with GreedyNMS, on various datasets and convolutional object de-

Table 2. Comparisons of our method and GreedyNMS on KITTI dataset, with ResNet variants as the backbone of Faster-RCNN.

Method	mAP(easy to hard)			Car			Pedestrian			Cyclist		
				Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mode	Hard
GreedyNMS	94.7	89.8	83.0	99.1	97.0	87.4	90.2	81.5	74.8	94.8	90.5	86.9
Ours (ResNet-50)	93.4	88.5	82.8	96.4	95.6	87.9	90.1	80.9	74.7	93.6	89.0	85.7
GreedyNMS	94.0	88.6	81.5	98.6	95.9	86.4	89.7	81.4	74.1	93.7	88.7	84.1
Ours (ResNet-101)	93.5	88.1	81.2	95.9	95.5	86.1	89.5	79.5	72.1	95.1	89.1	85.2
GreedyNMS	94.6	89.8	83.0	98.3	95.7	86.2	91.0	83.2	76.7	94.4	90.5	86.1
Ours (ResNet-152)	93.8	89.5	82.7	96.8	96.1	86.9	90.7	82.8	75.6	93.8	89.5	85.6

Table 3. Comparisons of our method and GreedyNMS on PASCAL VOC dataset, with both two-stage detector Faster-RCNN and one-stage detector SSD.

Detection Pipeline	GreedyNMS	Ours
Faster-RCNN [21] (ResNet-50)	78.1	77.6
Faster-RCNN [21] (ResNet-101)	78.4	78.0
Faster-RCNN [21] (ResNet-152)	78.7	78.4
SSD [19] (VGG-16)	77.3	76.1
SSD [19] (MobileNet-v1)	67.6	66.4
SSD [19] (MobileNet-v2)	68.7	67.8

tectors. First, we perform experiments on KITTI dataset with Faster-RCNN detector and report detection results in Table 2. We observe that our method achieves comparable detection accuracy with GreedyNMS on KITTI at most of the operating points, regardless of the backbone models used. Second, we perform experiments on PASCAL VOC dataset with both two-stage (*i.e.* , Faster-RCNN) and one-stage (*i.e.* , SSD) detectors. As shown in Table 3, one can see that our approach performs slightly worse (less than 1% for most of the operating points) than GreedNMS with various backbone models and different object detectors. With Faster-RCNN detector and ResNet-152 as the backbone, the performance gap in mAP between PSRR-MaxpoolNMS and GreedyNMS is only 0.3%. It is also worth noting that our PSRR-MaxpoolNMS approach is applicable to various object detection pipelines.

4.4. Efficiency

We perform both theoretical and experimental analysis on the computing efficiency of our method PSRR-MaxpoolNMS. Table 4 provides the theoretical analysis of the time complexity for GreedyNMS and our method. Given the number of input boxes N, the time complexity for both the Relationship Recovery and Pyramid Shifted MaxpoolNMS is $\mathcal{O}(N)$, which is much smaller than that of the GreedyNMS $\mathcal{O}(N \log N) + \mathcal{O}(N^2)$. Moreover, both the Relationship Recovery and Pyramid Shifted MaxpoolNMS can be easily parallelized, which in turn would further reduce the execution time of PSRR-MaxpoolNMS.

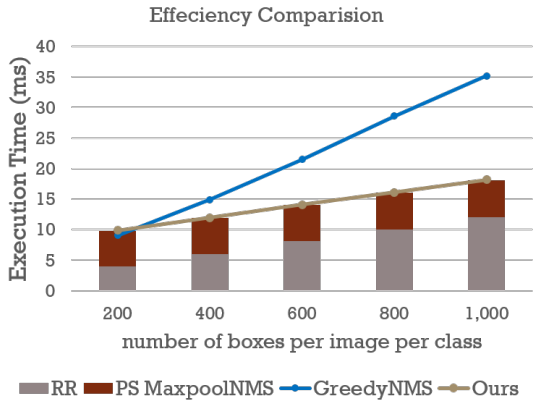

Figure 6. Execution time (in ms) of our method and GreedyNMS as a function of the number of bounding boxes being processed. We also report the timing breakdown of our method (Relationship Recovery (RR), and Pyramid Shifted (PS) MaxpoolNMS). Both methods run on CPU.

We also measure the execution time of GreedyNMS and our method on Intel(R) Core(TM) i9-10900X CPU, with different number of bounding boxes being processed. We experiment on SSD with VGG-16 as backbone on PASCAL VOC dataset. For fair comparison, we remove the score thresholding step in order to set the number of input boxes being processed per image per class. Results are reported in Figure 6. First, with the increasing number of input bounding boxes, our PSRR-MaxpoolNMS is more and more efficient than GreedyNMS. When the number of box is increased to 8000, the excutation time of PSRR-MaxpoolNMS is 89 ms while the GreedyNMS takes 512 ms, which is almost 6 times slower than our method. Second, we look into the timing breakdown of PSRR-MaxpoolNMS. We observe that the execution time of Pyramid Shifted MaxpoolNMS is almost constant, while the execution time of Relationship Recovery linearly increases with the number of boxes, due to our own implementation of Relationship Recovery is currently not parallelized. For Pyramid Shifted MaxpoolNMS, we rely on the PyTorch Maxpool and MaxUnpool API which are already parallelized on CPU.

Table 4. Time Complexity and Parallelism.

Method	Complexity	Parallel
Relationship Recovery	$\mathcal{O}(N)$	✓
PS MaxpoolNMS	$\mathcal{O}(N)$	✓
GreedyNMS	$\mathcal{O}(N \log N) + \mathcal{O}(N^2)$	×

Table 5. Effect of each component of Pyramid MaxpoolNMS on Spatial and Channel Recovery.

-	baseline	spa recover	spa + cha recover
single	41.2	44.4	71.1
ratio	49.0	63.0	73.6
scale	31.3	67.5	68.0
all	38.9	63.8	63.9

4.5. Ablation studies

In this section, we perform ablation studies to evaluate different components in our method, i.e., Relationship Recovery and Pyramid Shifted MaxpoolNMS. All experimental results are reported on PASCAL VOC dataset, with ResNet-50 as the backbone of Faster-RCNN detector.

4.5.1 Relationship Recovery

Spatial and Channel Recovery. We analyze the effect of each component of Pyramid MaxpoolNMS on spatial and channel recovery. Results are reported in Table 5. One can see that compared with the baseline that is lacking of recovered relationships (because it is based on the anchor box projection), both the spatial and channel recovery step could alleviate the score map mismatch problem and improve the detection accuracy over the baseline by a large margin, regardless of the channel combination used in the subsequent max pooling stage.

Score Assignment. As mentioned before, for score assignment in each cell of the score maps, we only keep the box with the highest score, which can be treated as a pre-filtering step based on max pooling in each cell (max-assign). We also investigate alternatives beyond max-assign, i.e., random-assgin and sum-assign. 'random-assign' is to randomly choose a box (and its score value) that projected to a cell. 'sum-assign' is to sum the score values of all boxes projected to a cell. Like max-assign, both random-assign and sum-assign could be easily parallelized. Table 6 reports the detection mAP with different score assignment variants. We observe that max-assign performs the best, followed by sum-assign and random-assign.

4.5.2 Pyramid Shifted MaxpoolNMS

Pyramid MaxpoolNMS. We evaluate the effect of the number of channel combinations in a sequence and the execution order in the sequence that defined by Pyramid MaxpoolNMS. Table 7 reports the detection results. First, single-scan max pooling (*i.e.* , single, ratio, scale, all.) performs consistently worse than multi-scan max pooling predefined in a sequence (e.g., a sequence single+ratio has 2 max pooling), implying the necessity of Pyramid MaxpoolNMS. Second, if we execute the sequence in reverse order (e.g. for a sequence single+ratio, execute ratio first, followed by single), the performance is slightly worse than the original execution order.

Table 6. Effect of the score assignment variants on Relationship Recovery.

method	random-assign	sum-assign	max-assign
mAP (%)	75.1	76.5	77.6

Table 7. Effect of the channel combinations and the execution order in the sequence of the Pyramid MaxpoolNMS.

-	original order	reverse order
single (Single-Scale)	71.1	-
ratio (Cross-Ratio)	73.6	-
scale (Cross-Scale)	68.0	-
all (Cross-All-Channel)	63.9	-
single+ratio	74.2	73.7
single+ratio+all	76.9	76.1
single+ratio+scale+all	77.6	77.4

Table 8. Effect of the Shifted MaxpoolNMS.

-	w/o shift-pool	w/ shift-pool
mAP (%)	74.2	77.6

Shifted MaxpoolNMS for Edge Effect. We perform study on our Shifted MaxpoolNMS and report the results in Table 8. It shows that without the additional Shifted MaxpoolNMS to alleviate the edge effect, the detection mAP drops obviously about 3.4%.

5. Conclusion

In this paper, we propose PSRR-MaxpoolNMS as a parallelizable alternative to GreedyNMS for overlapped box removal in all convolutional object detectors. With the proposed Relationship Recovery module and the Pyramid Shifted MaxpoolNMS module, we tackle the problems of score map mismatch and low sparsity after pooling on the score maps. Comprehensive experiments show that PSRR-MaxpoolNMS achieves comparable detection accuracy with GreedyNMS, but with much higher speedup in execution time.

References

[1] 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. 3

[2] Lile Cai, Bin Zhao, Zhe Wang, Jie Lin, Chuan Sheng Foo, Mohamed Sabry Aly, and Vijay Chandrasekhar. Maxpool-nms: getting rid of nms bottlenecks in two-stage object detectors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9356–9364, 2019. 1, 2, 3, 5, 6

[3] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016. 1

[4] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005. 3

[5] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6569–6578, 2019. 5

[6] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 6

[7] Nils Gählert, Niklas Hanselmann, Uwe Franke, and Joachim Denzler. Visibility guided nms: Efficient boosting of amodal object detection in crowded traffic scenes. *arXiv preprint arXiv:2006.08547*, 2020. 3

[8] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 6

[9] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 1, 2

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

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[13] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. A convnet for non-maximum suppression. In *German Conference on Pattern Recognition*, pages 192–204. Springer, 2016. 3

[14] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 6

[15] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th Annual International Symposium on Computer Architecture*, pages 1–12, 2017. 1

[16] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 734–750, 2018. 5

[17] 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. 2

[18] Songtao Liu, Di Huang, and Yunhong Wang. Adaptive nms: Refining pedestrian detection in a crowd. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6459–6468, 2019. 3

[19] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 1, 2, 6, 7

[20] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 1, 2

[21] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 1, 2, 6, 7

[22] Niels Ole Salscheider. Featurenms: Non-maximum suppression by learning feature embeddings. *arXiv preprint arXiv:2002.07662*, 2020. 3

[23] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 6

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

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