RMOPP: Robust Multi-Objective Post-Processing for Effective Object Detection

Mayuresh Savargaonkar    Abdallah Chehade
Department of Industrial and Manufacturing Systems Engineering
University of Michigan-Dearborn

Samir Rawashdeh
Department of Electrical and Computer Engineering
University of Michigan-Dearborn

mayuresh@umich.edu    achehade@umich.edu    srawa@umich.edu

Abstract

Over the last few decades, many architectures have been developed that harness the power of neural networks to detect objects in near real-time. Training such systems requires substantial time across multiple GPUs and massive labeled training datasets. Although the goal of these systems is generalizability, they are often impractical in real-life applications due to flexibility, robustness, or speed issues. This paper proposes RMOPP: A robust multi-objective post-processing algorithm to boost the performance of fast pre-trained object detectors with a negligible impact on their speed. Specifically, RMOPP is a statistically driven, post-processing algorithm that allows for simultaneous optimization of precision and recall. A unique feature of RMOPP is the Pareto frontier that identifies dominant possible post-processed detectors to optimize for both precision and recall. RMOPP explores the full potential of a pre-trained object detector and is deployable for near real-time predictions. We also provide a compelling test case on YOLOv2 using the MS-COCO dataset.

1. Introduction

Object detectors and vision systems have greatly advanced in the last few years. The evolution of deep neural networks has unlocked new potential in complex fields such as capacity estimations of Li-ion battery cells [36, 33, 6] and Remaining-Useful-Life (RUL) predictions [35, 4, 5]. The field of object detection and classification is another significant beneficiary of this especially due to the advancements in Convolutional Neural Networks (CNNs). In this paper, we refer to the broader task of detection and classification simply as object detection. Deep convolution neural networks commonly outperform traditional methods such as ones using Support Vector Machines (SVMs) or regression techniques [24]. CNNs are typically either run as a single-stage detector or a two-stage detector. More details on state-of-art detectors are provided in the literature review.

Research today mainly focuses on building object detection systems that offer a reasonable trade-off between speed and accuracy. For improved accuracy, two-stage detectors including ensemble neural networks like PA Net [25] are shown to perform well in competitions like PASCAL VOC [27], MS-COCO [23], and ImageNet [7]. Although such systems provide competition winning performances, they sacrifice speed and are not currently suitable for deployment in real-time systems. For improved latency, single-stage detectors like the YOLO series algorithms: YOLOv1 [30], YOLOv2 [31], and YOLOv3 [10] are shown to be faster than two-stage detectors. Although single-stage detectors are extremely fast, they often fail to reach the same level of robustness as the two-stage detectors [24]. Robustness here is defined as the ability of an object detection pipeline to successfully handle perturbations. Single-stage detectors’ limitations are often associated with localization errors and the identification of false positives [29]. This makes them less suitable for critical applications that require high robustness.

Practically, for many real-life applications like object detection in autonomous vehicles in Figure 1, both, detection of an object and speed are deemed extremely important. A failure in detection, as well as a slow detection, can result in a fatal crash. Further, these object detection pipelines are often hard-wired to achieve high precision at lower recall or higher recall at lower precision to achieve competition winning performances. Therefore, there is an essential need for a customizable and low-latency object detection pipeline with a robust post-processing method that can pro-
cess at real-time speeds without undergoing re-training.

Generally, it is hard to tune a robust object detection pipeline with a uni-objective post-processing algorithm. While existing methods focus on optimizing either precision/recall they suffer from maintaining an acceptable recall/precision. In this paper, we propose RMOPP: A robust multi-objective post-processing algorithm that allows for simultaneous optimization of both, precision and recall. RMOPP exhibits the following advantages:

- **Statistical Guarantees** - Development of a heuristic based on the log-likelihood ratio provides explainable statistical reasoning.

- **Increased Robustness** - Intuitive post-processing hyper-parameters developed using statistical guarantees allow for robust and improved filtering.

- **Pareto Frontier** - Development of a multi-dimensional post-processing algorithm allows for a thorough exploration of an object detection pipeline using the concepts of Pareto frontier. This helps in the choice of optimized post-processing hyper-parameters. More details on the Pareto frontier are discussed in Section 4.

- **Need for Re-training** - The need for re-training the object detector is eliminated when preference-based optimization for Precision and/or Recall is required.

- **Decreased Complexity** - Avoids design, development, and training of complex post-processing procedures while achieving state-of-art performances.

- **Low Latency** - Since the post-processing hyper-parameters are optimized offline, the inference time of the object detection pipeline remains unaffected.

Although we present a case study based on the single-stage detectors in this paper, there is no reason to assume that a similar change in the post-processing step won’t be beneficial for the two-stage detectors. The remainder of the paper is organized as follows: Section 2 explores literature in the field of object detection and post-processing, Section 3 introduces the proposed post-processing algorithm, Section 4 verifies the efficacy of the proposed algorithm using a case-study and Section 5 concludes the paper with directions for future work.

2. Literature Review

2.1. Object Detectors: State-of-Art

Modern object detectors can be categorized into two types: (i) Two-stage detectors like R-CNN [13], Fast R-CNN [12], Faster R-CNN [32], and many others, (ii) Single-stage detectors like OverFeat [34], YOLO [30], RetinaNet [22] and many others. The family of R-CNN type two-stage object detectors works on selective search strategies or Region Proposal Networks (RPN) [13]. The first stage proposes and filters locations inside the image that have a high probability of containing an object, and the second stage scores these proposed regions. Fast R-CNN reduces the computational load in the first stage by passing the image only once through a convolutional layer that generates Regions of Interest (ROI) using the selective search strategy. Faster R-CNN improves upon Fast R-CNN by using an RPN instead of a selective search strategy which further offers a 10x improvement in speed [32]. Cascade R-CNN [2] uses multiple R-CNN modules capable of detecting varying sizes of objects with different classification thresholds for each module. Frameworks such as Libra-RCNN [28] work on performance improvements by working on the imbalance in training images. To offer speed improvements over RPN stages of Fast R-CNN, a novel method, CPNDet [9] was recently published by Duan et al. CPNDet replaces the RPN stage with an anchor-free heat-map based detection framework called as CornerNet [20]. Although two-stage detectors show better detection performance, they offer limited improvements in speed due to several complex modules being stitched together to produce final detections. Moreover, these detectors have far too many hyper-parameters which make them unsuitable for real-life applications.

Single-stage detectors are fundamentally different and try to achieve the entire process of detection, classification, and bounding box regression using a single step. Most single-stage detectors work extremely fast but fail to reach the same level of robustness as two-stage detectors. Algorithms such as DSSD [11], SSD [26], and YOLO [31], [10] use anchor-based multi-scale detections for identifying fine-grained features in the image and thus improve the accuracy of the detector. Object detectors like RetinaNet [22] try to improve on the localization errors created by the single-stage detectors by introducing ‘focal loss’ that rewards the model under training when it correctly classifies the hard-negative examples. To eliminate problems associated with detections of small-scale objects in common single-stage detectors, researchers recently introduced detectors such as CenterNet [8] and FCOS [40]. These detectors work by eliminating the need for anchor boxes. In their detailed ex-
Experiments with ATSS, Zhang et al. [43] show that definitions of positive and negative annotations help bridge traditional gaps between anchor-free and anchor-based methods.

End-to-end object detection is another branch of single-stage detectors that try to improve performance by eliminating the post-processing stages such as NMS (Non-Maximal Suppression). DETR [3] is the first such detector that is capable of performing end-to-end object detection. OneNet [38] is another recent end-to-end object detector that achieves state-of-art while running at real-time speeds. While end-to-end object detectors achieve unparalleled performances in speed and accuracy, they lack customizability offered by post-processing stages. Customizations require re-training using several GPUs and hours of complex training paradigms often unavailable to the common user. The importance of this subject is also touched upon by J. Huang et al. [18], although they focus more on exploiting the aspect of speed and accuracy trade-offs for developing a better detector. While investigating the effect of speed on accuracy for a wide range of detectors, they show that the single-stage detectors based on the ResNet 101 backbone perform equally well when compared to R-FCN [18] and Faster RCNN on the same backbone for large object detection on the MS-COCO dataset.

2.2. Post-Processing Algorithms: State-of-Art

Most existing object detectors start with outputting four bounding box co-ordinates, a score, and the class associated with each one of the detections. The commonly used ‘score’ function for detection is the product of the probability of the most probable class bounded in an identified detection and the confidence of having an object inside the detected box. Next, detections with low scores are filtered assuming that lower scores represent poorly identified or classified objects. Once the detections are filtered, NMS is applied with a pre-defined NMS threshold ($\eta$) [1]. NMS is the process of identifying detections with the Intersection over Union (IoU) greater than the NMS threshold ($\eta$) and then eliminating those with lower ‘scores’. The NMS process is effective in eliminating redundant detections.

A significant improvement in post-processing comes as the ‘Soft-NMS’ replacing the traditional greedy NMS [1]. This work has shown to be of little use in the case of complex datasets like the PASCAL VOC and MS-COCO [1]. To propose a suitable improvement over the ‘Soft-NMS’, He et al. [15] introduced the ‘Softer-NMS’ which decays the bounding box scores using a continuous function thus achieving better results. Relation networks [17] and Learning NMS [16] are amongst some other types of works that try to improve upon the traditional NMS by designing and training a sub-network to analyze complex object-object correlations. While works such as Fitness NMS [41] try to integrate localization information into ranking scores, LTR [39] is another sub-network that improves on suppression ranking via learning procedures. NMS based post-processing works show promising performance gains and thus help build confidence that effective post-processing is key in any suitable object detection pipeline. A summary of a common single-stage object detection pipeline is shown in Figure 2.

Although there exist post-processing algorithms, they often fail to explicitly acknowledge the fact that there may exist multiple recalls at the same precision and vice-versa. Thus, they fail to explore the full potential of an object de-
3. Proposed Algorithm: RMOPP

Existing post-processing algorithms rely on a single data-driven ‘score’ metric that is not statistically intuitive. Several assumptions are made when filtering detections by scores and NMS using pre-defined post-processing hyper-parameters and thresholds. These values are usually chosen on an empirical basis, user’s experience, or by using a trial-and-error method. Hereafter, we will focus on the most common ‘score’ function for existing post-processing algorithms that is defined by the product of the probability of the most probable class bounded in an identified detection and the confidence of having an object inside the detected bounding box.

With this definition of the score function, Table 1 summarizes the output of commonly used object detectors. Block I comprise detections with poor scores that are typically filtered out. Block IV comprises of detections with high scores that are typically retained. Block II and III comprise of risky detections with intermediate scores. The existing post-processing algorithms are susceptible to poor performance for detections in Block II and III that have intermediate levels scores. Many of those detections are (i) mistakenly filtered out (a large number of false negatives) thus achieving a lower than ideal recall or, (ii) poor detections are retained (a large number of false positives) thus achieving a lower than ideal precision. To address this limitation, we introduce RMOPP: A robust multi-objective, post-processing algorithm that allows simultaneous and optimized tuning of recall and precision.

### 3.1. Proposed Post-Processing Algorithm

Let \( X^c \) be the binary random variable associated with the existence of some bounding box \( c \) with probability \( P_c \). Let \( X_i^c \) be the binary random variable indicating that the object in the bounding box \( c \) belongs to a class \( i \) with probability \( P_i^c \). Under the assumptions that \( X^c \sim Bern(P_c) \) and the collection \( \{X_i^c, ..., X_i^c, ..., X_N^c \} \sim \text{MultiNom}(P_i^c, ..., P_N^c, 1) \),

\[
E[X^c] = P(X^c = 1) = P_c
\]

\[
E[X_i^c] = P(X_i^c = 1) = P_i^c
\]

\[
\sum_{i=1}^{N} P_i^c = 1
\]

To simplify the mathematical notations, let \( Z_i^c \) be the ordered statistic of \( (X_i^c) \) such that \( P(Z_i^c = 1) \times P(Z_j^c = 1) \forall j > i \). Under this notation, \( P(Z_i^c = 1) \) is the probability of the most probable class for the detection bounded in the box \( c \) and the existing post-processing technique is expressed as,

\[
s^c = P(Z_1^c = 1) \times P(X^c = 1) \leq \gamma
\]

where the detection in the box \( c \) is filtered out if its score \( s^c \) is less than a pre-defined threshold \( \gamma \) as stated in Equation 4. The threshold \( \gamma \) is often chosen based on empirical analysis, user experience, or by limited trials. Extensions of the of Equation 4 exist where the score is defined as \( \max_i P(X_i^c = 1) \) or as \( \max_i P(X^c = 1) \). It should be noted that if the object detector does not output the confidence \( P(X^c = 1) \), then \( P(X^c = 1) \) is set to 1 for all detections.

Note that during training, object detectors aim to minimize the population cross-entropy over the training dataset. Therefore, it is natural to design a post-processing algorithm that shares similar merit to the training procedure. Specifically, we consider thresholding over the log-likelihood ratios of the most probable class to all other classes for every bounding box. This brings us to our first proposition.

**Proposition I**: The log-likelihood ratios between the top class and the remaining classes of a bounding box \( c \) are sufficient to quantify the classification accuracy of the object in the bounding box \( c \).

Proposition I is inspired by the likelihood ratio test and the theory of statistical hypothesis testing, which shows that every bounding box \( c \) must satisfy

\[
\frac{P(Z_i^c = 1) \times P(Z_j^c = 1) \forall j > i}{P(Z_i^c = 1) \times P(Z_j^c = 1) \forall j > i} \geq \gamma_1, \forall k > 1.
\]

**Lemma I**: Given the definition that \( (Z_i^c) \) is the ordered statistic of \( (X_i^c) \) such that \( P(Z_i^c = 1) \times P(Z_j^c = 1) \forall j > i \). Then, \( \frac{P(Z_i^c = 1) \times P(Z_j^c = 1) \forall j > i}{P(Z_i^c = 1) \times P(Z_j^c = 1) \forall j > i} \geq \gamma_1 \) for any bounding box \( c \) guarantees \( \frac{P(Z_i^c = 1) \times P(Z_j^c = 1) \forall j > i}{P(Z_i^c = 1) \times P(Z_j^c = 1) \forall j > i} \geq \gamma_1, \forall k > 1. \) From Lemma I and Proposition I, we identify our first score metric to be

\[
\frac{P(Z_i^c = 1) \times P(Z_j^c = 1) \forall j > i}{P(Z_i^c = 1) \times P(Z_j^c = 1) \forall j > i} \geq \gamma_1, \forall k > 1.
\]
and its user-defined threshold to be \( \gamma_1 \). Therefore, any detected box \( (c) \) must satisfy Equation 5 or it will be removed for poorly classifying its bounded object.

\[
\left( \frac{P(Z_c^1 = 1)}{P(Z_2^1 = 1)} \right) \geq \gamma_1, \forall c
\]

(5)

While the classification task is critical in object detection, the bounding box detection is also an important task. Proposition I guarantees a good classification but it does not guarantee a good bounding box detection. This brings us to Proposition II.

**Proposition II:** An effective object detector must show a relatively similar classification and detection performance which can be quantified by the log-likelihood ratio of detection to classification for any bounding box \( (c) \),

\[
\left( \frac{P(X^c = 1)}{P(Z_2^1 = 1)} \right), \forall k > 1.
\]

Proposition II suggests that a detected bounding box truly bounds an object if \( \left( \frac{P(X^c = 1)}{P(Z_2^1 = 1)} \right), \forall k > 1 \), is large enough. It should be noted here that there exists a cross-correlation between Propositions I and II, where Proposition II mainly focuses on the relative detection performance given the classification probabilities \( P\{X^c = 1\}P(Z_2^1 = 1) \), ..., \( P(Z_k^1 = 1) \) for any detected bounding box \( (c) \), thus ensuring that the boxes are well-identified and classified.

**Lemma II:** Given the definition that \( (Z_i^c) \) is the ordered statistic of \( (X_i^c) \) such that \( P(Z_i^c = 1) \times P(Z_j^c = 1) \forall j > i \). Then, \( \left( \frac{P(X^c = 1)}{P(Z_2^1 = 1)} \right) \geq \gamma_2 \) for any bounding box \( (c) \), guarantees \( \left( \frac{P(X^c = 1)}{P(Z_2^1 = 1)} \right) \geq \gamma_2, \forall k > 1 \). From Lemma II and Proposition II, we identify our second score metric to be \( \left( \frac{P(X^c = 1)}{P(Z_2^1 = 1)} \right) \) and its user-defined threshold to be \( \gamma_2 \). Therefore, any detected box \( (c) \) must satisfy Equation 6 or it will be considered a false detection and removed.

\[
\left( \frac{P(X^c = 1)}{P(Z_2^1 = 1)} \right) \geq \gamma_2, \forall c
\]

(6)

Both Equations 5 and 6 summarize the proposed post-processing algorithm, where a bounding box \( (c) \) is a good detection if and only if,

\[
\left( \frac{P(Z_i^c = 1)}{P(Z_2^1 = 1)} \right) \geq \gamma_1 \ cap \left( \frac{P(X^c = 1)}{P(Z_2^1 = 1)} \right) \geq \gamma_2, \forall c
\]

(7)

### 3.2. Advantages of RMOPPP

Equation 5 reduces the chances of having false positives due to poor classification. This addresses the problems with Block III in Table 1 for traditional post-processing algorithms because it filters out detections with low \( \left( \frac{P(X^c = 1)}{P(Z_2^1 = 1)} \right) \).

Note that a lower value for \( \left( \frac{P(X^c = 1)}{P(Z_2^1 = 1)} \right) \) indicates poor classification based on Lemma I and Proposition I. Thus, it is hypothesized that increasing the value \( \gamma_1 \) will reduce false positives due to poor classification. Note that extreme values for \( \gamma_1 \) may have a detrimental effect on true positives, thus increasing the false negatives and lowering precision and hence, should be chosen carefully. More details and results are presented in Section 4.

Further, Lemma II helps address problems due to cases following Block II in Table 1. Intuitively, \( P(Z_2^1 = 1) \), should only be considered when \( P(X^c = 1) \) is high enough. At random detections where \( P(X^c = 1) \) is low and \( P(Z_2^1 = 1) \) is very high, we can observe that the traditional algorithms fail as they eliminate too many true positives. Thus, it is hypothesized that a lower \( \gamma_2 \) will help increasing true positives by keeping detections that were traditionally eliminated due to poor identification. The correlation between \( \gamma_1 \) and \( \gamma_2 \) will be particularly helpful here in keeping the false positives at bay. Figure 3 shows how results improve when RMOPPP is used for filtering the detections compared to the traditional algorithms.

### 3.3. Selecting Hyper-Parameters \( \gamma_1 \) and \( \gamma_2 \)

Choosing \( \gamma_1 \) and \( \gamma_2 \) is a non-trivial task that depends on the object detector’s raw output. This is further elaborated in the case-study. Generally, decreasing both \( \gamma_1 \) and \( \gamma_2 \) will result in filtering out fewer detections.
Algorithm 1: Pseudocode for RMOPP

Require: \( \left\{ \arg \max_i X_i^c, Z_i^c, Z_i^e, X_i^e, b_i^c \right\} \) \( M \) predictions for a given image with box coord \( b_i^c \);
\( \delta_1, \delta_2 \) are the increments for \( \gamma_1, \gamma_2 \);
for \( \Delta_i^1 \leq \gamma_2 \leq \Delta_i^U \) do
for \( \Delta_i^1 \leq \gamma_1 \leq \Delta_i^U \) do
\( I = 0 \); an indicator if the detections satisfy Eq. 7
for \( c = 1, \ldots, M \) do
if \( \left( \frac{P(Z_i^c=1)}{P(Z_i^e=1)} \right) \geq \gamma_1 \) and \( \left( \frac{P(X_i^c=1)}{P(X_i^e=1)} \right) \geq \gamma_2 \) then
\( I[c] = 1 \)
end if
end for
if \( \sum(I) \neq 0 \) then
apply NMS with IoU = \( \eta \)
compute Precision, Recall and \( F_1 \)
end if
for \( \gamma_1 = \gamma_1 + \delta_1 \) end for
for \( \gamma_2 = \gamma_2 + \delta_2 \) end for
return Precision, Recall and \( F_1 \) scores \( \forall \gamma_1 \) and \( \gamma_2 \)

In summary, the object detection pipeline with RMOPP performs the following steps: (1.) Take an input image and pass it through the pre-trained detector that outputs some detections, (2.) Filter these detections using Equation 7, given some values for (\( \gamma_1 \)) and (\( \gamma_2 \)), (3.) Perform NMS on detections from step 2 to further refine the results using NMS threshold (\( \eta \)), (4.) Calculate precision, recall, and \( F_1 \) scores, (5.) Modify values of (\( \gamma_1 \)) and (\( \gamma_2 \)) and repeat steps (2-5), (6.) Choose the best values of (\( \gamma_1 \)) and (\( \gamma_2 \)) based on preference for precision, recall, and/or \( F_1 \) scores. It should also be noted that since there is no formal specification of the NMS threshold (\( \eta \)), we default it to 0.5 for all experiments unless specified otherwise.

4. Case Study: YOLOv2 using Darknet-19

While RMOPP is a suitable add-on for most of the available object detectors, we consider applying it to improve YOLOv2 in this case-study. YOLOv2 is one of the fastest existing object detectors; however, it detects many erroneous boxes and that makes it an excellent case-study for evaluating the efficacy of post-processing algorithms.

YOLOv2’s backbone is called as ‘Darknet-19’ that consists of 19 hidden convolutional layers, and it was first introduced in 2017 [31]. It is one of the fastest architectures around and performs about 5.58 billion operations per image [31]. The YOLOv2 loss function is a multi-part loss function which is a weighted sum of classification, localization, and confidence losses [30]. The ‘classification loss’ represents the loss based on the squared error of conditional class probabilities of the detected object given by the softmax layer of the detector. The ‘localization loss’ represents the loss based on the difference between coordinates of the bounding boxes for detected objects and the coordinates of bounding boxes in annotations. Note that a weighting parameter (\( \lambda_{coord} \)) is used to increase the contribution of localization loss in the overall loss function. ‘Confidence loss’ represents the loss concerning the objectness score for each bounding box. Since we expect more boxes for background class, to eliminate the class imbalance, another weighting parameter (\( \lambda_{noobj} \)) is used to down weight the contribution of confidence loss in the overall loss function when a bounding box is identified as a background object. (\( \lambda_{coord} \)) and (\( \lambda_{noobj} \)) are set to 5 and 0.5 by the Redmon et al. in [31].

The MS-COCO dataset used for training YOLOv2 consists of around 118,000 images with an average of 7 boxes per image and a total of 80 unique class labels [23]. For performance evaluation, we use the COCO minival dataset which has 5000 images that were not used in training the YOLOv2, and set the input image resolution to 544x544 for all our experiments.

4.1. Results: Precision, Recall, and \( F_1 \) score

In this case study, we use the definitions of precision, recall, and \( F_1 \) score as given in Equations 8, 9 and 10. A detection is considered as a true positive if the (IoU) is greater than 0.5 with at least one ground truth annotation of the same class label. Note that if multiple objects have (IoU) more than 0.5, then only the one with the highest (IoU) is selected as a true positive while others are considered to be false positives. All unmatched objects in the list of ground
truths are considered to be false negatives. For this case-study \((\gamma_1)\) is varied between 1 and 10 with increments of 0.5 and \((\gamma_2)\) is varied between 0.1 and 1 with increments of 0.05. Beyond those limits, YOLOv2 showed extremely poor precisions or recalls.

\[
\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{8}
\]

\[
\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{9}
\]

\[
F_1 \text{ score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}
\]

Changes in \((\gamma_1)\) and \((\gamma_2)\) have a significant effect on precision, recall, and \(F_1\) scores. Essentially, \((\gamma_1)\) and \((\gamma_2)\) can be perceived as tuning knobs that can be used to tune the object detector for desired precision, recall, or \(F_1\) score. In Figure 4a it is seen that as \((\gamma_1)\) is increased, a smooth decrease in recall is observed with recall almost dropping to 0 for the highest values of \((\gamma_2)\). Similarly, in Figure 4b it is observed that as \((\gamma_2)\) is increased, precision increases almost linearly to 0.9 or more when \((\gamma_2)\) is augmented from 0.1 to 1 for all values of \((\gamma_1)\) above 5.5. For values of \((\gamma_1)\) lower than 5.5 this increase is not so significant due to the correlation between \((\gamma_1)\) and \((\gamma_2)\).

When \((\gamma_1)\) and \((\gamma_2)\) are set to highest values, total detections drop, and a maximum precision condition (Recall≈0) is realized. Similarly, when \((\gamma_1)\) and \((\gamma_2)\) are set to lowest values, total detections rise and a maximum recall condition (Precision≈0) is realized. This helps build a sense of confidence which suggests that by employing finer adjustments in \((\gamma_1)\) and \((\gamma_2)\), we can explore the entire possible space of precision-recall offline. In this case study, multiple post-processed detectors with different \((\gamma_1)\) and \((\gamma_2)\) resulted in the same precision but different recall (and vice versa). The region highlighted in Figure 4a shows possible combinations of \((\gamma_1)\) and \((\gamma_2)\) where recall is fixed to 0.5. The regions highlighted in Figure 4b show possible combinations of \((\gamma_1)\) and \((\gamma_2)\) where precision is fixed to 0.5.

Designing a bi-objective function to simultaneously optimize precision and recall is challenging, thus, researchers have developed metrics like \(F_1\) score to address this challenge. In Figure 4c we plot a heat map of \(F_1\) scores in Figure 4c by varying \((\gamma_1)\) and \((\gamma_2)\). Although Figure 4c helps us understand the complex correlations between \((\gamma_1)\), \((\gamma_2)\), precision and recall, it offers limited help in selecting values of \((\gamma_1)\) and \((\gamma_2)\) that maximize precision at given recall and vice-versa. Thus, we plot a Pareto frontier using results obtained by varying \((\gamma_1)\) and \((\gamma_2)\). The Pareto frontier is plotted as a function of precision and recall in Figure 5. Here, we use the concepts of ‘Pareto Optimality’ [19] and extend them to the field of object detection for the

Figure 4: Effects of \((\gamma_1)\) and \((\gamma_2)\) on Precision, Recall and \(F_1\) score.
first time, to the best of our knowledge. In object detection, ‘Pareto Optimality’ is defined as a condition where no improvements can be made to either precision or recall without some sacrifice in another. Such combinations of \((\gamma_1)\) and \((\gamma_2)\) which achieve an optimal trade-off between precision and recall are called ‘Pareto Optimal’ points. A set of such Pareto optimal points is called as ‘Pareto Frontier’. Pareto optimal points are said to dominate non-optimal points because there exists no other combination of \((\gamma_1)\) and \((\gamma_2)\) which achieves a better precision (or recall) at a higher recall (or precision). Figure 5 shows a plot of each unique combination of \((\gamma_1)\) and \((\gamma_2)\) and their associated precision and recall as a solid ‘black’ dot. A Pareto frontier is highlighted in ‘red’ in this figure which helps choose the best possible combinations of \((\gamma_1)\) and \((\gamma_2)\) that maximize precision at given recall or vice-versa. Figure 5, thus, can be used to thoroughly explore and select a robust choice of hyper-parameters \((\gamma_1)\) and \((\gamma_2)\).

Although Pareto Frontier helps in choosing the best combinations of \((\gamma_1)\) and \((\gamma_2)\), setting values for \((\gamma_1)\) and \((\gamma_2)\) that maximize precision (or recall) without ensuring some minimum recall (or precision) is not recommended as it will result in too few or too many detections, which is not practical. Thus, to avoid this, we set a minimum threshold on the F\(_1\) score to be 0.5 in this case study. The optimal hyper-parameters achieving maximum precision, recall, and F\(_1\) score under this condition are thus pinpointed using dominance principles in the Pareto frontier using Figure 5. Table 2 summarizes these values.

| Maximizing Objective | \(F_1\) score | Recall | Precision | \(\gamma_1\) | \(\gamma_2\) |
|----------------------|---------------|--------|-----------|-------------|-------------|
| Precision            | 0.50          | 0.35   | 0.88      | 10          | 0.55        |
| Recall               | 0.50          | 0.60   | 0.43      | 3.5         | 0.15        |
| \(F_1\) score        | 0.57          | 0.49   | 0.68      | 4.5         | 0.35        |

Table 2: Pareto Optimal \((\gamma_1)\) and \((\gamma_2)\) under different maximization objectives.

4.2. Comparison with Benchmark Methods

For benchmark comparisons, we utilize the COCO AP metric that has been extensively used to evaluate various state-of-art detectors on the COCO challenge [23]. This metric is based on the PASCAL VOC 2012 metric which used a similar strategy to evaluate object detectors [27]. Both VOC 2012 and COCO metrics calculate the per class Average Precision or (AP) by calculating the area under the Precision-Recall curve. This procedure is repeated for different (IoU) between 0.5 and 0.95. The individual AP for every (IoU) is reported as AP (IoU) and the mean AP is reported as AP [0.5:0.95].

Table 3 summarizes the comparisons between the results of RMOPP and other benchmarked post-processing methods. Using Table 3, it can be observed that RMOPP under different optimal settings significantly improves the COCO AP metrics for YOLOv2 without any loss in inference times (FPS). Specifically, for AP [0.5:0.95], RMOPP shows a 29\% increase compared to the original benchmark of YOLOv2 [31]. Henceforth, we mainly focus on AP [0.5] because for practical applications the detected boxes must have at least a 50\% intersection-over-union with the ground truth or it will be considered a bad detection. Using Table 3, it can be observed that, the proposed method easily surpasses the performance by other benchmark post-processing methods such as Soft-NMS [1] and RelationNets [17] while being significantly faster on an NVIDIA Titan X GPU. Here it should be noted that although Rank-NMS [39] achieves better performance, it is significantly slower than the proposed method. Figures 6 and 7 compares performance of YOLOv2 with traditional filtering and RMOPP. The compelling evidence in these figures helps prove the effectiveness of RMOPP over traditional algorithms.

4.3. Comparison to State-of-Art Object Detectors

Table 4 aims to compare the performance of the YOLOv2 with RMOPP to other state-of-art object detectors [14, 22, 38]. The AP [0.5] results in Table 4 show that RMOPP improves the YOLOv2 performance to become comparable to that of its slower but more complex successor- YOLOv3. It also shows that better post-processing can significantly improve the performance of single-stage detectors like YOLOv2 to match that of a two-
| Method          | Backbone   | Post-Processing Method | FPS | $\gamma_1$ | $\gamma_2$ | AP [0.5]  | AP [0.5:0.95] | AP [0.75] |
|-----------------|------------|------------------------|-----|------------|------------|------------|---------------|------------|
| YOLOv2 [31]     | Darknet-19 | NMS(Baseline)          | 40  | -          | -          | 0.216      | 0.440         | 0.192      |
| MetanAnchorGS [42] | Darknet-19 | NMS                    | -   | -          | -          | 0.212      | 0.395         | -          |
| Faster R-CNN [17] | ResNet-50  | Soft-NMS               | 9   | -          | -          | 0.300      | 0.523         | 0.305      |
| Faster R-CNN [17] | ResNet-50  | RelationNets           | 17  | -          | -          | 0.303      | 0.519         | 0.315      |
| Faster R-CNN [39] | ResNet-50-FPN | Rank-NMS          | 8   | -          | -          | 0.386      | **0.582**     | **0.417**  |
| YOLOv2          | Darknet-19 | RMOPP-Best AP         | 40  | 1          | 0.1        | 0.278      | **0.524**     | 0.266      |
| YOLOv2          | Darknet-19 | RMOPP-Best Precision  | 40  | 10         | 0.55       | 0.216      | 0.366         | 0.229      |
| YOLOv2          | Darknet-19 | RMOPP-Best Recall     | 40  | 3.5        | 0.15       | 0.268      | 0.502         | 0.258      |
| YOLOv2          | Darknet-19 | RMOPP-Best $F_1$ score| 40  | 4.5        | 0.35       | 0.253      | 0.459         | 0.251      |

Table 3: Comparison of RMOPP across benchmarks using the COCO metrics.

| Method          | $\gamma_1$ | $\gamma_2$ | AP [0.5]  |
|-----------------|------------|------------|------------|
| **Two-Stage Detectors** |           |                 |            |
| Faster R-CNN+++ [14] | -          | -          | 0.557      |
| Faster R-CNN w FPN [21] | -          | -          | 0.591      |
| Faster R-CNN w D-RMI [18] | -          | -          | 0.555      |
| Faster R-CNN w TDM [37] | -          | -          | 0.577      |
| **Single-Stage Detectors** |           |                 |            |
| YOLOv3 608x608 [10] | -          | -          | 0.579      |
| SSD513 [11]      | -          | -          | 0.504      |
| DSSD513 [11]     | -          | -          | 0.533      |
| RetinaNet+ResNet-101-FPN [22] | -          | -          | 0.591      |
| RetinaNet+ResNetXT-101-FPN [22] | -          | -          | **0.611**  |
| CenterNet+ResNet-18 [38] | -          | -          | 0.466      |
| OneNet+ResNet-18 [38] | -          | -          | 0.457      |
| YOLOv2+RMOPP-Best AP | 1          | 0.1        | **0.524**  |
| YOLOv2+RMOPP-Best Recall | 3.5       | 0.15       | **0.502**  |
| YOLOv2+RMOPP-Best $F_1$ score | 4.5        | 0.35       | **0.459**  |
| YOLOv2+RMOPP-Best Precision | 10       | 0.55       | **0.366**  |

Table 4: Comparison of RMOPP with state-of-art.

stage detector like Faster R-CNN.

5. Conclusion and Futurework

This paper introduces RMOPP: A robust multi-objective post-processing algorithm that improves the performance of pre-trained object detectors with a negligible impact on speed. Unlike existing uni-objective post-processing methods, the proposed algorithm allows for simultaneous optimization of precision and recall in object detection pipelines. When applied to YOLOv2, RMOPP showed a 29% improvement in average precision to match similar performances of slower but more complex object detectors like YOLOv3 and Faster-RCNN. This work presents the first known usage of Pareto frontiers for post-processing hyper-parameter optimization in the field of object detec-

Figure 6: YOLOv2+RMOPP shows significant improvement in results.

...
Figure 7: YOLOv2+RMOPP shows significant improvement in results.

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.

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: Delving into High Quality Object Detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, volume 2015, pages 1440–1448, 2015.

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In European Conference on Computer Vision, pages 213–229. Springer, 2020.

[4] Abdallah Chehade, Scott Bonk, and Kaibo Liu. Sensory-based failure threshold estimation for remaining useful life prediction. IEEE Transactions on Reliability, 66(3):939–949, 2017.

[5] Abdallah Chehade and Zunya Shi. Sensor Fusion via Statistical Hypothesis Testing for Prognosis and Degradation Analysis. IEEE Transactions on Automation Science and Engineering, 16(4):1774–1787, 2019.

[6] Abdallah A Chehade and Ala A Hussein. Latent Function Decomposition for Forecasting Li-ion Battery Cells Capacity: A Multi-Output Convolved Gaussian Process Approach, 2019.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 248–255, 2010.

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

[9] Kaiwen Duan, Lingxi Xie, Honggang Qi, Song Bai, Qiming Huang, and Qi Tian. Corner proposal network for anchor-free, two-stage object detection. arXiv preprint arXiv:2007.13816, 2020.

[10] Ali Farhadi and Joseph Redmon. Yolov3: An incremental improvement. Computer Vision and Pattern Recognition, cite as, 2018.

[11] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C Berg. Dssd: Deconvolutional single shot detector. arXiv preprint arXiv:1701.06659, 2017.

[12] Ross Girshick. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, volume 2015, pages 1440–1448, 2015.

[13] Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, U C Berkeley, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, volume 1, page 5000, 2014.

[14] 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.

[15] Yihui He, Chenchen Zhu, Jianren Wang, Marios Savvides, and Xiangyu Zhang. Bounding box regression with uncertainty for accurate object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2888–2897, 2019.

[16] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. Learning non-maximum suppression. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4507–4515, 2017.

[17] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3588–3597, 2018.

[18] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergey Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. In Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, volume 2017-Janua, pages 3296–3305, 2017.

[19] Gandhinath Karuppusami and R Gandhinathan. Pareto analysis of critical success factors of total quality management. The TQM magazine, 2006.
