Rank & Sort Loss for Object Detection and Instance Segmentation

Kemal Oksuz, Baris Can Cam, Emre Akbas*, Sinan Kalkan*
Dept. of Computer Engineering, Middle East Technical University, Ankara, Turkey
{kemal.oksuz, can.cam, eakbas, skalkan}@metu.edu.tr

Abstract

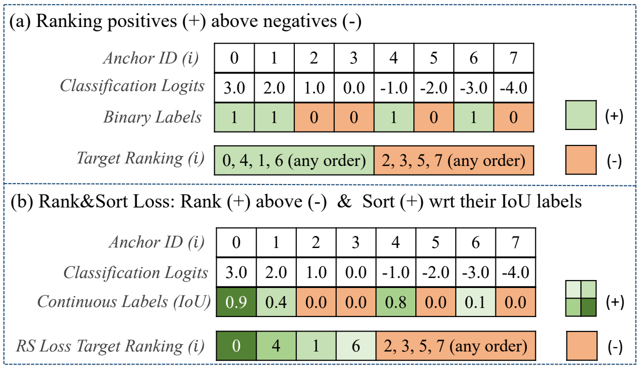
We propose Rank & Sort (RS) Loss, a ranking-based loss function to train deep object detection and instance segmentation methods (i.e. visual detectors). RS Loss supervises the classifier, a sub-network of these methods, to rank each positive above all negatives as well as to sort positives among themselves with respect to (wrt.) their localisation qualities (e.g. Intersection-over-Union - IoU). To tackle the non-differentiable nature of ranking and sorting, we reformulate the incorporation of error-driven update with backpropagation as Identity Update, which enables us to model our novel sorting error among positives. With RS Loss, we significantly simplify training: (i) Thanks to our sorting objective, the positives are prioritized by the classifier without an additional auxiliary head (e.g. for centerness, IoU, mask-IoU), (ii) due to its ranking-based nature, RS Loss is robust to class imbalance, and thus, no sampling heuristic is required, and (iii) we address the multi-task nature of visual detectors using tuning-free task-balancing coefficients. Using RS Loss, we train seven diverse visual detectors only by tuning the learning rate, and show that it consistently outperforms baselines: e.g. our RS Loss improves (i) Faster R-CNN by ~ 3 box AP and aLRP Loss (ranking-based baseline) by ~ 2 box AP on COCO dataset, (ii) Mask R-CNN with repeat factor sampling (RFS) by 3.5 mask AP (~ 7 AP for rare classes) on LVIS dataset; and also outperforms all counterparts. Code is available at: https://github.com/kemaloksuz/RankSortLoss.

1. Introduction

Owing to their multi-task (e.g. classification, box regression, mask prediction) nature, object detection and instance segmentation methods rely on loss functions of the form:

$$\mathcal{L}_{VD} = \sum_{k \in \mathcal{K}} \sum_{t \in \mathcal{T}} \lambda_t^k \mathcal{L}_t^k, \quad (1)$$

which combines $\mathcal{L}_t^k$, the loss function for task t on stage k (e.g. $|\mathcal{K}| = 2$ for Faster R-CNN [32] with RPN and R-CNN), weighted by a hyper-parameter λ_t^k. In such formulations, the number of hyper-parameters can easily exceed 10 [27], with additional hyper-parameters arising from task-specific imbalance problems [28], e.g. the positive-negative imbalance in the classification task, and if a cascaded architecture is used (e.g. HTC [7] employs 3 R-CNNs with different λ_t^k). Thus, although such loss functions have led to unprecedented successes, they require tuning, which is time consuming, leads to sub-optimal solutions and makes fair comparison of methods challenging.



(a) Ranking positives (+) above negatives (-)

Anchor ID (i)	0	1	2	3	4	5	6	7
Classification Logits	3.0	2.0	1.0	0.0	-1.0	-2.0	-3.0	-4.0
Binary Labels	1	1	0	0	1	0	1	0
Target Ranking (i)	0, 4, 1, 6 (any order)				2, 3, 5, 7 (any order)			

🟩 (+)
🟧 (-)

(b) Rank&Sort Loss: Rank (+) above (-) & Sort (+) wrt their IoU labels

Anchor ID (i)	0	1	2	3	4	5	6	7
Classification Logits	3.0	2.0	1.0	0.0	-1.0	-2.0	-3.0	-4.0
Continuous Labels (IoU)	0.9	0.4	0.0	0.0	0.8	0.0	0.1	0.0
RS Loss Target Ranking (i)	0	4	1	6	2, 3, 5, 7 (any order)			

🟩 (+)
🟧 (-)

Figure 1. A ranking-based classification loss vs RS Loss. (a) Enforcing to rank positives above negatives provides a useful objective for training, however, it ignores ordering among positives. (b) Our RS Loss, in addition to raking positives above negatives, aims to sort positives wrt. their continuous IoUs (positives: a green tone based on its label, negatives: orange). We propose Identity Update (Section 3), a reformulation of error-driven update with backpropagation, to tackle these ranking and sorting operations which are difficult to optimize due to their non-differentiable nature.

Recently proposed *ranking-based* loss functions, namely "Average Precision (AP) Loss" [6] and "average Localisation Recall Precision (aLRP) Loss" [27], offer two important advantages over the classical *score-based* functions (e.g. Cross-entropy Loss and Focal Loss [22]): (1) They directly optimize the performance measure (e.g. AP), thereby providing consistency between training and evaluation objectives. This also reduces the number of hyper-parameters as the performance measure (e.g. AP) does not typically have any hyper-parameters. (2) They are robust to class-

[*] Equal contribution for senior authorship.

imbalance due to their ranking-based error definition. Although these losses have yielded state-of-the-art (SOTA) performances, they need longer training and more augmentation.

Broadly speaking, the ranking-based losses (AP Loss and aLRP Loss) focus on ranking positive examples over negatives, but they do not explicitly model positive-to-positive interactions. However, there is evidence that it is helpful to prioritize predictions wrt. their localisation qualities by using an auxiliary (aux. - e.g. IoU, centerness) head [15, 17, 38, 44] or by supervising the classifier to directly regress IoUs of the predictions without an aux. head (as shown by Li et al. [18] in Quality Focal Loss - QFL).

In this paper, we propose Rank & Sort (RS) Loss as a ranking-based loss function to train visual detection (VD – i.e. object detection and instance segmentation) methods. RS Loss not only ranks positives above negatives (Fig. 1(a)) but also sorts positives among themselves with respect to their continuous IoU values (Fig. 1(b)). This approach brings in several crucial benefits. Due to the prioritization of positives during training, detectors trained with RS Loss do not need an aux. head, and due to its ranking-based nature, RS Loss can handle extremely imbalanced data (e.g. object detection [28]) without any sampling heuristics. Besides, except for the learning rate, RS Loss does not need any hyper-parameter tuning thanks to our tuning-free task-balancing coefficients. Owing to this significant simplification of training, we can apply RS Loss to different methods (i.e. multi-stage, one-stage, anchor-based, anchor-free) easily (i.e. *only by tuning the learning rate*) and demonstrate that RS Loss consistently outperforms baselines.

Our contributions can be summarized as follows:

(1) We reformulate the incorporation of error-driven optimization into backpropagation to optimize non-differentiable ranking-based losses as *Identity Update*, which uniquely provides interpretable loss values during training and allows definition of intra-class errors (e.g. the sorting error among positives).

(2) We propose *Rank & Sort Loss* that defines a ranking objective between positives and negatives as well as a sorting objective to prioritize positives wrt. their continuous IoUs. Due to this ranking-based nature, RS Loss can train models in the presence of highly imbalanced data.

(3) We present the effectiveness of RS Loss on a diverse set of four object detectors and three instance segmentation methods only by tuning the learning rate and without any aux. heads or sampling heuristics on the widely-used COCO and long-tailed LVIS benchmarks: E.g. (i) Our RS-R-CNN improves Faster-CNN by ~ 3 box AP on COCO, (ii) our RS-Mask R-CNN improves repeat factor sampling by ~ 3.5 mask AP (~ 7 AP for rare classes) on LVIS.

2. Related Work

Auxiliary heads and continuous labels. Predicting the localisation quality of a detection with an aux. centerness [38, 44], IoU [15, 17], mask IoU [14] or uncertainty head [13] and combining these predictions with the classification scores for NMS are shown to improve detection performance. Lin et al. [18] discovered that using continuous IoUs of predictions to supervise the classifier outperforms using an aux. head. Currently, Lin et al.'s "Quality Focal Loss" [18] is the only method that is robust to class imbalance [28] and uses continuous labels to train the classifier. With RS Loss, we investigate the generalizability of this idea on different networks (e.g. multi-stage networks [2, 32]) and on a different task (i.e. instance segmentation).

Ranking-based losses in VD. Despite their advantages, ranking-based losses are non-differentiable and difficult to optimize. To address this challenge, black-box solvers [34] use an interpolated AP surface, though yielding little gain in object detection. DR Loss [31] achieves ranking between positives and negatives by enforcing a margin with Hinge Loss. Finally, AP Loss [6] and aLRP Loss [27] optimize the performance metrics, AP and LRP [26] respectively, by using the error-driven update of perceptron learning [35] for the non-differentiable parts. However, they need longer training and heavy augmentation. The main difference of RS Loss is that it also considers continuous localisation qualities as labels.

Objective imbalance in VD. The common strategy in VD is to use λ_t^k (Eq. 1), a scalar multiplier, on each task and tune them by grid search [1, 17]. Recently, Oksuz et al. [27] employed a self-balancing strategy to balance classification and box regression heads, both of which compete for the bounded range of aLRP Loss. Similarly, Chen et al. [5] use the ratio of classification and regression losses to balance these tasks. In our design, each loss $\mathcal{L}_t^k$ for a specific head has its own bounded range and thus, no competition ensues among heads. Besides, we use $\mathcal{L}_t^k$s with similar ranges, and show that our RS Loss can simply be combined with a simple task balancing strategy based on loss values, and hence does not require any tuning except the learning rate.

3. Identity Update for Ranking-based Losses

Using a ranking-based loss function is attractive thanks to its compatibility with common performance measures (e.g. AP). It is challenging, however, due to the non-differentiable nature of ranking. Here, we first revisit an existing solution [6, 27] that overcomes this non-differentiability by incorporating error-driven update [35] into backpropagation (Section 3.1), and then present our reformulation (Section 3.2), which uniquely (i) provides interpretable loss values and (ii) takes into account intra-class errors, which is crucial for using continuous labels.

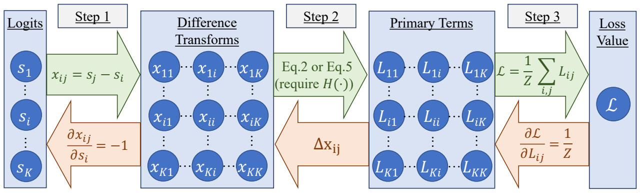


Figure 2. Three-step computation (green arrows) and optimization (orange arrows) algorithms of ranking-based loss functions. Our identity update (i) yields interpretable loss values (see Supp.Mat. for an example on our RS Loss), (ii) replaces Eq. 2 of previous work [27] by Eq. 5 (green arrow in Step 2) to allow intra-class errors, crucial to model our RS Loss, and (iii) results in a simple "Identity Update" rule (orange arrow in Step 2): $\Delta x_{ij} = L_{ij}$.

3.1. Revisiting the Incorporation of Error-Driven Optimization into Backpropagation

Definition of the Loss. Oksuz et al. [27] propose writing a ranking-based loss as $\mathcal{L} = \frac{1}{Z} \sum\limits_{i \in \mathcal{P}} \ell(i)$ where Z is a problem specific normalization constant, $\mathcal{P}$ is the set of positive examples and $\ell(i)$ is the error term computed on $i \in \mathcal{P}$.

Computation of the Loss. Given logits (s_i), $\mathcal{L}$ can be computed in three steps [6, 27] (Fig. 2 green arrows):

Step 1. The difference transform between logits s_i and s_j is computed by $x_{ij} = s_j - s_i$.

Step 2. Using x_{ij}, errors originating from each pair of examples are calculated as primary terms (L_{ij}):

$$L_{ij} = \begin{cases} \ell(i)p(j|i), & \text{for } i \in \mathcal{P}, j \in \mathcal{N} \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where $p(j|i)$ is a probability mass function (pmf) that distributes $\ell(i)$, the error computed on $i \in \mathcal{P}$, over $j \in \mathcal{N}$ where $\mathcal{N}$ is the set of negative examples. By definition, the ranking-based error $\ell(i)$, and thus L_{ij}, requires pairwise-binary-ranking relation between outputs i and j, which is determined by the non-differentiable unit step function $\mathrm{H}(x)$ (i.e. $\mathrm{H}(x) = 1$ if $x \geq 0$ and $\mathrm{H}(x) = 0$ otherwise) with input x_{ij}.

Using $\mathrm{H}(x_{ij})$, different ranking-based functions can be introduced to define $\ell(i)$ and $p(j|i)$: e.g. the rank of the ith example, $\mathrm{rank}(i) = \sum\limits_{j \in \mathcal{P} \cup \mathcal{N}} \mathrm{H}(x_{ij})$; the rank of the ith example among positives, $\mathrm{rank}^{+}(i) = \sum\limits_{j \in \mathcal{P}} \mathrm{H}(x_{ij})$; and number of false positives with logits larger than s_i, $\mathrm{N_{FP}}(i) = \sum\limits_{j \in \mathcal{N}} \mathrm{H}(x_{ij})$. As an example, for AP Loss [6], using these definitions, $\ell(i)$ and $p(j|i)$ can be simply defined as $\frac{\mathrm{N_{FP}}(i)}{\mathrm{rank}(i)}$ and $\frac{\mathrm{H}(x_{ij})}{\mathrm{N_{FP}}(i)}$ respectively [27].

Step 3. Finally, $\mathcal{L}$ is calculated as the normalized sum of the primary terms [27]: $\mathcal{L} = \frac{1}{Z} \sum\limits_{i \in \mathcal{P}} \ell(i) = \frac{1}{Z} \sum\limits_{i \in \mathcal{P}} \sum\limits_{j \in \mathcal{N}} L_{ij}$.

Optimization of the Loss. Here, the aim is to find updates $\frac{\partial \mathcal{L}}{\partial s_i}$, and then proceed with backpropagation through model parameters. Among the three computation steps (Fig. 2 orange arrows), Step 1 and Step 3 are differentiable, whereas a primary term L_{ij} is not a differentiable function of difference transforms. Denoting this update in x_{ij} by Δx_{ij} and using the chain rule, $\frac{\partial \mathcal{L}}{\partial s_i}$ can be expressed as:

$$\frac{\partial \mathcal{L}}{\partial s_i} = \sum_{j,k} \frac{\partial \mathcal{L}}{\partial L_{jk}} \Delta x_{jk} \frac{\partial x_{jk}}{\partial s_i} = \frac{1}{Z} \Big(\sum_{j} \Delta x_{ji} - \sum_{j} \Delta x_{ij} \Big). \tag{3}$$

Chen et al. [6] incorporate the error-driven update [35] and replace Δx_{ij} by $-(L_{ij}^{*} - L_{ij})$ where L_{ij}^{*} is the target primary term indicating the desired error for pair (i, j). Both AP Loss [6] and aLRP Loss [27] are optimized this way.

3.2. Our Reformulation: Identity Update

We first identify two drawbacks of the formulation in Section 3.1: (D1) Resulting loss value ($\mathcal{L}$) does not consider the target L_{ij}^{*}, and thus, is not easily interpretable when $L_{ij}^{*} \neq 0$ (cf. aLRP Loss [27] and our RS Loss - Section 4); (D2) Eq. 2 assigns a non-zero primary term only if $i \in \mathcal{P}$ and $j \in \mathcal{N}$, effectively ignoring intra-class errors. These errors become especially important with continuous labels: The larger the label of $i \in \mathcal{P}$, the larger should s_i be.

Definition of the Loss. We redefine the loss function as:

$$\mathcal{L} = \frac{1}{Z} \sum_{i \in \mathcal{P} \cup \mathcal{N}} (\ell(i) - \ell^{*}(i)), \tag{4}$$

where $\ell^{*}(i)$ is the desired error term on $i \in \mathcal{P}$. Our loss definition has two benefits: (i) $\mathcal{L}$ directly measures the difference between the target and the desired errors, yielding an interpretable loss value to address (D1), and (ii) we do not constrain $\mathcal{L}$ to be defined only on positives and replace "$i \in \mathcal{P}$" with "$i \in \mathcal{P} \cup \mathcal{N}$". Although we do not use "$i \in \mathcal{P} \cup \mathcal{N}$" to model RS Loss, it makes the definition of $\mathcal{L}$ complete in the sense that, if necessary to obtain $\mathcal{L}$, individual errors ($\ell(i)$) can be computed on each output, and hence, $\mathcal{L}$ can be approximated more precisely or a larger set of ranking-based loss functions can be represented.

Computation of the Loss. In order to compute $\mathcal{L}$ (Eq. 4), we only replace Eq. 2 with:

$$L_{ij} = \big(\ell(i) - \ell^{*}(i)\big)\, p(j|i), \tag{5}$$

in three-step algorithm (Section 3.1, Fig. 2 green arrows) and allow all pairs to have a non-zero error, addressing (D2).

Optimization of the Loss. Since the error of a pair, L_{ij}, is minimized when $\ell(i) = \ell^{*}(i)$, Eq. 5 has a target of $L_{ij}^{*} = 0$ regardless of $\mathcal{L}$. Thus, Δx_{ij} in Eq. 3 is simply the primary term itself: $\Delta x_{ij} = -(L_{ij}^{*} - L_{ij}) = -(0 - L_{ij}) = L_{ij}$, concluding the derivation of our *Identity Update*.

4. Rank & Sort Loss

In order to supervise the classifier of visual detectors by considering the localisation qualities of the predictions (e.g.

IoU), RS Loss decomposes the problem into two tasks: (i) *Ranking task*, which aims to rank each positive higher than all negatives, and (ii) *sorting task*, which aims to sort the logits s_i in descending order wrt. continuous labels y_i (e.g. IoUs). We define RS Loss and compute its gradients using our Identity Update (Section 3.2 – Fig. 2).

Definition. Given logits s_i and their continuous labels $y_i \in [0,1]$ (e.g. IoU), we define RS Loss as the average of the differences between the current ($\ell_{\mathrm{RS}}(i)$) and target ($\ell^*_{\mathrm{RS}}(i)$) RS errors over positives (i.e. $y_i > 0$):

$$\mathcal{L}_{\mathrm{RS}} := \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \left(\ell_{\mathrm{RS}}(i) - \ell^*_{\mathrm{RS}}(i) \right), \tag{6}$$

where $\ell_{\mathrm{RS}}(i)$ is a summation of the current ranking error and current sorting error:

$$\ell_{\mathrm{RS}}(i) := \underbrace{\frac{\mathrm{N_{FP}}(i)}{\mathrm{rank}(i)}}_{\ell_{\mathrm{R}}(i)\text{: Current Ranking Error}} + \underbrace{\frac{\sum\limits_{j \in \mathcal{P}} \mathrm{H}(x_{ij})(1-y_j)}{\mathrm{rank}^+(i)}}_{\ell_{\mathrm{S}}(i)\text{: Current Sorting Error}}. \tag{7}$$

For $i \in \mathcal{P}$, while the "current ranking error" is simply the precision error, the "current sorting error" penalizes the positives with logits larger than s_i by the average of their inverted labels, $1-y_j$. Note that when $i \in \mathcal{P}$ is ranked above all $j \in \mathcal{N}$, $\mathrm{N_{FP}}(i) = 0$ and target ranking error, $\ell^*_{\mathrm{R}}(i)$, is 0. For target sorting error, we average over the inverted labels of $j \in \mathcal{P}$ with larger logits ($\mathrm{H}(x_{ij})$) and labels ($y_j \geq y_i$) than $i \in \mathcal{P}$ corresponding to the desired sorted order,

$$\ell^*_{\mathrm{RS}}(i) = \cancelto{0}{\ell^*_{\mathrm{R}}(i)} + \underbrace{\frac{\sum\limits_{j \in \mathcal{P}} \mathrm{H}(x_{ij})[y_j \geq y_i](1-y_j)}{\sum\limits_{j \in \mathcal{P}} \mathrm{H}(x_{ij})[y_j \geq y_i]}}_{\ell^*_{\mathrm{S}}(i)\text{:Target Sorting Error}}, \tag{8}$$

where $[\mathrm{P}]$ is the Iverson Bracket (i.e. 1 if predicate P is True; else 0), and similar to previous work [6], $\mathrm{H}(x_{ij})$ is smoothed in the interval $[-\delta_{RS}, \delta_{RS}]$ as $x_{ij}/2\delta_{RS} + 0.5$.

Computation. We follow the three-step algorithm (Section 3, Fig. 2) and define primary terms, L_{ij}, using Eq. 5, which allows us to express the errors among positives as:

$$L_{ij} = \begin{cases} \left(\ell_{\mathrm{R}}(i) - \ell^*_{\mathrm{R}}(i)\right) p_R(j|i), & \text{for } i \in \mathcal{P}, j \in \mathcal{N} \\ \left(\ell_{\mathrm{S}}(i) - \ell^*_{\mathrm{S}}(i)\right) p_S(j|i), & \text{for } i \in \mathcal{P}, j \in \mathcal{P}, \\ 0, & \text{otherwise}, \end{cases} \tag{9}$$

where ranking ($p_R(j|i)$) and sorting pmfs ($p_S(j|i)$) uniformly distribute ranking and sorting errors on i respectively over examples causing error (i.e. for ranking, $j \in \mathcal{N}$ with $s_j > s_i$; for sorting, $j \in \mathcal{P}$ with $s_j > s_i$ but $y_j < y_i$):

$$p_R(j|i) = \frac{\mathrm{H}(x_{ij})}{\sum\limits_{k \in \mathcal{N}} \mathrm{H}(x_{ik})}; p_S(j|i) = \frac{\mathrm{H}(x_{ij})[y_j < y_i]}{\sum\limits_{k \in \mathcal{P}} \mathrm{H}(x_{ik})[y_k < y_i]}, \tag{10}$$

Optimization. To obtain $\frac{\partial \mathcal{L}_{\mathrm{RS}}}{\partial s_i}$, we simply replace Δx_{ij} (Eq. 3) by the primary terms of RS Loss, L_{ij} (Eq. 9), following Identity Update (Section 3.2). The resulting $\frac{\partial \mathcal{L}_{\mathrm{RS}}}{\partial s_i}$ for $i \in \mathcal{N}$ then becomes (see Supp.Mat. for derivations):

$$\frac{\partial \mathcal{L}_{\mathrm{RS}}}{\partial s_i} = \frac{1}{|\mathcal{P}|} \sum_{j \in \mathcal{P}} \ell_{\mathrm{R}}(j) p_R(i|j). \tag{11}$$

Owing to the additional sorting error (Eq. 7, 8), $\frac{\partial \mathcal{L}_{\mathrm{RS}}}{\partial s_i}$ for $i \in \mathcal{P}$ includes update signals for both promotion and demotion to sort the positives accordingly:

$$\frac{1}{|\mathcal{P}|} \Bigg(\underbrace{\ell^*_{\mathrm{RS}}(i) - \ell_{\mathrm{RS}}(i)}_{\text{Update signal to promote } i} + \underbrace{\sum_{j \in \mathcal{P}} \left(\ell_{\mathrm{S}}(j) - \ell^*_{\mathrm{S}}(j)\right) p_S(i|j)}_{\text{Update signal to demote } i} \Bigg). \tag{12}$$

Note that the directions of the first and second part of Eq. 12 are different. To place $i \in \mathcal{P}$ in the desired ranking, $\ell^*_{\mathrm{RS}}(i) - \ell_{\mathrm{RS}}(i) \leq 0$ promotes i based on the error computed on itself, whereas $\left(\ell_{\mathrm{S}}(j) - \ell^*_{\mathrm{S}}(j)\right) p_S(i|j) \geq 0$ demotes i based on the signal from $j \in \mathcal{P}$. We provide more insight for RS Loss and its gradients on an example in Supp.Mat.

5. Using RS Loss to Train Visual Detectors

This section develops an overall loss function to train detectors with RS Loss, in which only the learning rate needs tuning. As commonly performed in the literature [17, 18], Section 5.2 analyses different design choices on ATSS [44], a SOTA one-stage object detector (i.e. $k = 1$ in Eq. 1); and Section 5.3 extends our design to other architectures.

5.1. Dataset and Implementation Details

Unless explicitly specified, we use (i) standard configuration of each detector and only replace the loss function, (ii) mmdetection framework [8], (iii) 16 images with a size of 1333×800 in a single batch (4 images/GPU, Tesla V100) during training, (iv) $1\times$ training schedule (12 epochs), (v) single-scale test with images with a size of 1333×800, (vi) ResNet-50 backbone with FPN [21], (vii) COCO *trainval35K* (115K images) and *minival* (5k images) sets [23] to train and test our models, (iix) report COCO-style AP.

5.2. Analysis and Tuning-Free Design Choices

ATSS [44] with its classification, box regression and centerness heads is originally trained by minimizing:

$$\mathcal{L}_{ATSS} = \mathcal{L}_{cls} + \lambda_{box}\mathcal{L}_{box} + \lambda_{ctr}\mathcal{L}_{ctr}, \tag{13}$$

where $\mathcal{L}_{cls}$ is Focal Loss [22]; $\mathcal{L}_{box}$ is GIoU Loss [33]; $\mathcal{L}_{ctr}$ is Cross-entropy Loss with continuous labels to supervise centerness prediction; and $\lambda_{box} = 2$ and $\lambda_{ctr} = 1$. We first remove the centerness head and replace $\mathcal{L}_{cls}$ by our RS Loss

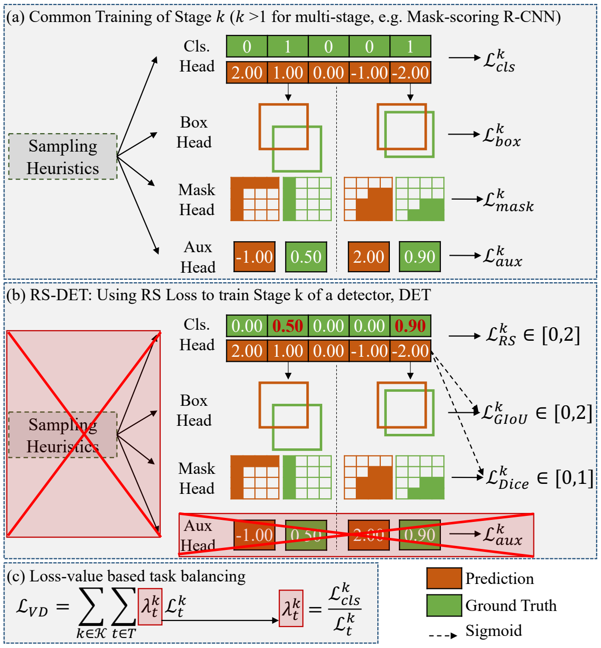

(a) Common Training of Stage k (k >1 for multi-stage, e.g. Mask-scoring R-CNN)
Cls. Head
0 1 0 0 1
2.00 1.00 0.00 -1.00 -2.00
$\mathcal{L}^k_{cls}$
Sampling Heuristics
Box Head
$\mathcal{L}^k_{box}$
Mask Head
$\mathcal{L}^k_{mask}$
Aux Head
-1.00 0.50 2.00 0.90
$\mathcal{L}^k_{aux}$
(b) RS-DET: Using RS Loss to train Stage k of a detector, DET
Cls. Head
0.00 0.50 0.00 0.00 0.90
2.00 1.00 0.00 -1.00 -2.00
$\mathcal{L}^k_{RS} \in [0,2]$
Sampling Heuristics
Box Head
$\mathcal{L}^k_{GIoU} \in [0,2]$
Mask Head
$\mathcal{L}^k_{Dice} \in [0,1]$
Aux Head
-1.00 0.50 2.00 0.90
$\mathcal{L}^k_{aux}$
(c) Loss-value based task balancing
$\mathcal{L}_{VD} = \sum_{k \in \mathcal{K}} \sum_{t \in T} \lambda^k_t \mathcal{L}^k_t$
$\lambda^k_t = \frac{\mathcal{L}^k_{cls}}{\mathcal{L}^k_t}$
Prediction
Ground Truth
Sigmoid

Figure 3. (a) A generic visual detection pipeline includes many heads from possibly multiple stages. An aux. head, in addition to the standard ones, is common in recent methods (e.g. centerness head for ATSS [44], IoU head for IoU-Net [15], and mask IoU head for Mask-scoring R-CNN [14]) to regress localisation quality and prioritize examples during inference (e.g. by multiplying classification scores by the predicted localisation quality). Sampling heuristics are also common to ensure balanced training. Such architectures use many hyper-parameters and are sensitive for tuning. (b) Training detectors with our RS Loss removes (i) aux. heads by directly supervising the classification (Cls.) head with continuous IoUs (in red & bold), (ii) sampling heuristics owing to its robustness against class imbalance. We use losses with similar range with our RS Loss in other branches (i.e. GIoU Loss, Dice Loss) also by weighting each by using classification scores, obtained applying sigmoid to logits. (c) Instead of tuning λ^k_ts, we simply balance tasks by considering loss values. With this design, we train several detectors only by tuning the learning rate and improve their performance consistently.

(Section 4), $\mathcal{L}_{RS}$, using $\text{IoU}(\hat{b}_i, b_i)$ between a prediction box ($\hat{b}_i$) and ground truth box (b_i) as the continuous labels:

$$\mathcal{L}_{RS-ATSS} = \mathcal{L}_{RS} + \lambda_{box}\mathcal{L}_{box}, \tag{14}$$

where λ_{box}, the *task-level balancing coefficient*, is generally set to a constant scalar by grid search.

Inspired by recent work [5, 27], we investigate two tuning-free heuristics to determine λ_{box} every iteration: (i) value-based: $\lambda_{box} = \mathcal{L}_{RS}/\mathcal{L}_{box}$, and (ii) magnitude-based: $\lambda_{box} = \left|\frac{\partial \mathcal{L}_{\text{RS}}}{\partial \hat{\mathbf{s}}}\right| / \left|\frac{\partial \mathcal{L}_{\text{box}}}{\partial \hat{\mathbf{b}}}\right|$ where $|\cdot|$ is L1 norm, $\hat{\mathbf{b}}$ and $\mathbf{s}$ are box regression and classification head outputs respectively. In our analysis on ATSS trained with RS Loss, we observed that value-based task balancing performs similar to tuning λ_{box} ($\sim$ 0 AP difference on average). Also, we use score-based weighting [18] by multiplying the GIoU Loss of each prediction using its classification score (details are in Supp.Mat.). Note that value-based task balancing and score-based instance weighting are both hyper-parameter-free and easily applicable to all networks. *With these design choices, Eq. 14 has only* 1 *hyper-parameter* (i.e. δ_{RS} in $\text{H}(\cdot)$, set to 0.50, to smooth the unit-step function)

5.3. Training Different Architectures

Fig. 3 presents a comparative overview on how we adopt RS Loss to train different architectures: When we use RS Loss to train the classifier (Fig. 3(b)), we remove aux. heads (e.g. IoU head in IoU-Net [15]) and sampling heuristics (e.g. OHEM in YOLACT [1], random sampling in Faster R-CNN [32]). We adopt score-based weighting in box regression and mask prediction heads, and prefer Dice Loss, instead of the common Cross-entropy Loss, to train mask prediction head for instance segmentation due to (i) its bounded range (between 0 and 1), and (ii) holistic evaluation of the predictions, both similar to GIoU Loss. Finally, we set λ^k_t (Eq. 1) to scalar $\mathcal{L}^k_{cls}/\mathcal{L}^k_t$ (i.e. $\mathcal{L}_{cls} = \mathcal{L}_{RS}$) every iteration (Fig. 3(c)) with the single exception of RPN where we multiply the losses of RPN by 0.20 following aLRP Loss.

6. Experiments

To present the contribution of RS Loss in terms of performance and tuning simplicity, we conduct experiments on seven visual detectors with a diverse set of architectures: four object detectors (i.e. Faster R-CNN [32], Cascade R-CNN [2], ATSS [44] and PAA [17] – Section 6.1) and three instance segmentation methods (i.e. Mask R-CNN [12], YOLACT [1] and SOLOv2 [40] – Section 6.2). Finally, Section 6.3 presents ablation analysis.

6.1. Experiments on Object Detection

6.1.1 Multi-stage Object Detectors

To train Faster R-CNN [32] and Cascade R-CNN [2] by our RS Loss (i.e. RS-R-CNN), we remove sampling from all stages (i.e. RPN and R-CNNs), use all anchors to train RPN and m top-scoring proposals/image (by default, $m = 1000$ for Faster R-CNN and $m = 2000$ Cascade R-CNN in mmdetection [8]), replace softmax classifiers by binary sigmoid classifiers and set the initial learning rate to 0.012.

RS Loss reaches 39.6 AP on a standard Faster R-CNN and outperforms (Table 1): (i) FPN [21] (Cross-entropy & Smooth L1 losses) by 3.4 AP, (ii) aLRP Loss [27], a SOTA ranking-based baseline, by 2.2 AP, (iii) IoU-Net [15] with aux. head by 1.5 AP and (iv) Dynamic R-CNN, closest counterpart, by 0.7 AP. We, then, use the lightweight Carafe [39] as the upsampling operation in FPN and obtain 40.8 AP (RS-R-CNN+), still maintaining $\sim$ 2 AP gap from Carafe

Table 1. RS-R-CNN uses the standard IoU-based assigner, is sampling-free, employs no aux. head, is almost tuning-free wrt. task-balancing weights (λ_t^ks – Eq. 1), and thus, has the least number of hyper-parameters (H# = 3 – two δ_{RS}, one for each RS Loss to train RPN & R-CNN, and one RPN weight). Still, RS-R-CNN improves standard Faster R-CNN with FPN by ~ 3 AP; aLRP Loss (ranking-based loss baseline) by ~ 2 AP; IoU-Net (a method with IoU head) by 1.5 AP. RS-R-CNN+ replaces upsampling of FPN by lightweight Carafe operation [39] and maintains ~ 2 AP gap from Carafe FPN (38.6 to 40.8 AP). All models use ResNet-50, are evaluated in COCO *minival* and trained for 12 epochs on mmdetection except for IoU-Net. H#: Number of hyper-parameters (Supp.Mat. presents details on H#.)

Method	Assigner	Sampler	Aux. Head	AP ↑	AP_{50} ↑	AP_{75} ↑	oLRP ↓	$\mathrm{oLRP}_{\mathrm{Loc}}$ ↓	$\mathrm{oLRP}_{\mathrm{FP}}$ ↓	$\mathrm{oLRP}_{\mathrm{FN}}$ ↓	H# ↓	Venue
FPN [21]	IoU-based	Random	None	36.5	58.5	39.4	70.1	18.3	27.8	45.8	9	CVPR 17
aLRP Loss [27]	IoU-based	None	None	37.4	57.9	39.2	69.2	17.6	28.5	46.1	**3**	NeurIPS 20
GIoU Loss [33]	IoU-based	Random	None	37.6	58.2	41.0	69.2	17.0	28.5	46.3	7	CVPR 19
IoU-Net [15]	IoU-based	Random	IoU Head	38.1	56.3	–	–	–	–	–	11	ECCV 18
Libra R-CNN [30]	IoU-based	IoU-based	None	38.3	59.5	41.9	68.8	17.2	27.5	45.4	11	CVPR 19
AutoLoss-A [24]	IoU-based	Random	None	38.5	58.6	41.8	68.4	16.6	27.1	45.5	7	ICLR 21
Carafe FPN [39]	IoU-based	Random	None	38.6	59.9	42.2	68.3	17.2	27.0	44.2	7	ICCV 19
Dynamic R-CNN [43]	Dynamic	Random	None	38.9	57.6	42.7	68.2	**15.7**	27.7	46.6	10	ECCV 20
RS-R-CNN (Ours)	IoU-based	None	None	39.6	59.5	43.0	67.9	16.3	27.8	45.4	**3**	
RS-R-CNN+ (Ours)	IoU-based	None	None	**40.8**	**61.4**	**43.8**	**66.9**	16.3	**26.4**	**43.7**	**3**	

FPN [39] (38.6 AP) and outperforming all methods in all AP- and oLRP-based [26, 29] performance measures except $\mathrm{oLRP}_{\mathrm{Loc}}$, which implies that our main contribution is in classification task trained by our RS Loss and there is still room for improvement in the localisation task. RS Loss also improves the stronger baseline Cascade R-CNN [2] by 1 AP from 40.3 AP to 41.3 AP (Supp.Mat. presents detailed results for Cascade R-CNN). Finally, RS Loss has the least number of hyper-parameters (H# = 3, Table 1) and does not need a sampler, an aux. head or tuning of λ_t^ks (Eq. 1).

6.1.2 One-stage Object Detectors

We train ATSS [44] and PAA [17] including a centerness head and an IoU head respectively in their architectures. We adopt the anchor configuration of Oksuz et al. [27] for all ranking-based losses (different anchor configurations do not affect performance of standard ATSS [44]) and set learning rate to 0.008. While training PAA, we keep the scoring function, splitting positives and negatives, for a fair comparison among different loss functions.

Comparison with AP and aLRP Losses, ranking-based baselines: We simply replaced Focal Loss by AP Loss to train networks, and as for aLRP Loss, similar to our RS Loss, we tuned its learning rate as 0.005 due to its tuning simplicity. Both for ATSS and PAA, RS Loss provides significant gains over ranking-based alternatives, which were trained for 100 epochs using SSD-like augmentation [25] in previous work [6, 27]: 1.8/2.2 AP gain for ATSS and 3.7/3.3 gain for PAA for AP/aLRP Loss (Table 2).

Comparison with Focal Loss, default loss function: RS Loss provides around ~ 1 AP gain when both networks are equally trained without an aux. head (Table 2) and 0.6 AP gain compared to the default networks with aux. heads.

Comparison with QFL, score-based loss function using continuous IoUs as labels: To apply QFL [18] to PAA, we remove the aux. IoU head (as we did with ATSS), test two possible options ((i) default PAA setting with $\lambda_{box} = 1.3$ and IoU-based weighting, (ii) default QFL setting: $\lambda_{box} = 2.0$ and score-based weighting) and report the best result for QFL. While the results of QFL and RS Loss are similar for ATSS, there is 0.8 AP gap in favor of our RS Loss, which can be due to the different positive-negative assignment method of PAA (Table 2).

6.1.3 Comparison with SOTA

Here, we use our RS-R-CNN since it yields the largest improvement over its baseline. We train RS-R-CNN for 36 epochs using multiscale training by randomly resizing the shorter size within $[480, 960]$ on ResNet-101 with DCNv2 [45]. Table 3 reports the results on COCO *test-dev*: Our RS-R-CNN reaches 47.8 AP and outperforms similarly trained Faster R-CNN and Dynamic R-CNN by ~ 3 and ~ 1 AP respectively. Although we do not increase the number of parameters for Faster R-CNN, RS R-CNN outperforms all multi-stage detectors including TridentNet [19], which has more parameters. Our RS-R-CNN+ (Section 6.1.1) reaches 48.2 AP, and RS-Mask R-CNN+ (Section 6.2) reaches 49.0 AP, outperforming all one- and multi-stage counterparts.

6.2. Experiments on Instance Segmentation

6.2.1 Multi-stage Instance Segmentation Methods

We train Mask R-CNN [12] on COCO and LVIS datasets by keeping all design choices of Faster R-CNN the same.

COCO: We observe ~ 2 AP gain for both segmentation and detection performance (Table 4) over Mask R-CNN. Also, RS-Mask R-CNN outperforms Mask-scoring R-CNN [14], with an additional aux. mask IoU head, by 0.4 mask AP, 1.8 box AP and 0.9 mask oLRP (Table 4).

LVIS: Replacing the Cross-entropy to train Mask R-CNN with repeat factor sampling (RFS) by our RS Loss improves the performance by 3.5 mask AP on the long-tailed LVIS dataset (21.7 to 25.2 with ~ $7\mathrm{AP}_{\mathrm{r}}$ improvement on rare classes) and outperforms recent counterparts (Table 5).

Table 2. RS Loss has the least number of hyper-parameters (H#) and outperforms (i) rank-based alternatives significantly, (ii) the default setting with an aux. head (underlined) by 0.6 AP, (iii) score-based alternative, QFL, especially on PAA. We test unified losses (i.e. a loss considering localisation quality while training classification head) only without aux. head. All models use ResNet-50.

Loss Function	Unified	Rank-based	Aux. Head	ATSS [44]				PAA [17]				H# ↓
				AP ↑	AP_{50} ↑	AP_{75} ↑	oLRP ↓	AP ↑	AP_{50} ↑	AP_{75} ↑	oLRP ↓	
Focal Loss [22]				38.7	57.6	41.5	68.9	39.9	57.3	43.4	68.7	3
			✓	39.3	57.5	42.6	68.6	40.4	58.4	43.9	67.7	4
AP Loss [6]		✓		38.1	58.2	41.0	69.2	35.3	53.1	38.5	71.5	2
		✓	✓	37.2	55.6	40.2	70.0	37.3	54.3	41.2	70.5	3
QFL [18]	✓			39.7	58.1	**42.7**	68.0	40.2	57.4	43.8	68.3	2
aLRP Loss [27]	✓	✓		37.7	57.4	39.9	69.4	37.7	56.1	40.1	69.9	**1**
RS Loss (Ours)	✓	✓		**39.9**	**58.9**	42.6	**67.9**	**41.0**	**59.1**	**44.5**	**67.3**	**1**

Table 3. Comparison with SOTA for object detection on COCO *test-dev* using ResNet-101 (except *) with DCN. The result of the similarly trained Faster R-CNN is acquired from Zhang et al. [43]. +: upsampling of FPN is Carafe [39], *:ResNeXt-64x4d-101

	Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
One-stage	ATSS [44]	46.3	64.7	50.4	27.7	49.8	58.4
	GFL [18]	47.3	66.3	51.4	28.0	51.1	59.2
	PAA [17]	47.4	65.7	51.6	27.9	51.3	60.6
	RepPointsv2 [10]	48.1	67.5	51.8	28.7	50.9	60.8
Multi-stage	Faster R-CNN [43]	44.8	65.5	48.8	26.2	47.6	58.1
	Trident Net [19]	46.8	67.6	51.5	28.0	51.2	60.5
	Dynamic R-CNN [43]	46.9	65.9	51.3	28.1	49.6	60.0
	D2Det [3]	47.4	65.9	51.7	27.2	50.4	61.3
Ours	RS-R-CNN	47.8	68.0	51.8	28.5	51.1	61.6
	RS-R-CNN+	48.2	68.6	52.4	29.0	51.3	61.7
	RS-Mask R-CNN+	**49.0**	**69.2**	**53.4**	**29.9**	**52.4**	**62.8**
	RS-Mask R-CNN+*	50.2	70.3	54.8	31.5	53.5	63.9

Table 4. Without an aux. head, RS-Mask R-CNN improves Mask R-CNN [12] by ~ 2 AP and outperforms Mask-scoring R-CNN [14] which employs an additional mask IoU head as aux. head.

Method	Aux Head	Segmentation Performance				AP_{box}	H# ↓
		AP ↑	AP_{50} ↑	AP_{75} ↑	oLRP ↓		
Mask R-CNN		34.7	55.7	37.2	71.2	38.2	8
Mask-sc. R-CNN	✓	36.0	55.8	38.7	71.0	38.2	9
RS-Mask R-CNN		**36.4**	**57.3**	**39.2**	**70.1**	**40.0**	**3**

Table 5. Comparison on LVIS v1.0 val set. Models are trained with ResNet-50, multiscale images (range: $[640, 800]$) for 12 epochs.

Method	AP_{mask}	AP_r	AP_c	AP_f	AP_{box}	Venue
RFS [11]	21.7	9.6	21.0	27.8	22.5	CVPR 19
BAGS [20]	23.1	13.1	22.5	28.2	23.7	CVPR 20
Eq. Lossv2 [37]	23.7	14.9	22.8	28.6	24.2	CVPR 21
RFS+RS Loss	**25.2**	**16.8**	**24.3**	**29.9**	**25.9**	

6.2.2 One-stage Instance Segmentation Methods

Here, we train two different approaches with our RS Loss: (i) YOLACT [1], a real-time instance segmentation method, involving sampling heuristics (e.g. OHEM [36]), aux. head and carefully-tuned loss weights, and demonstrate RS Loss can discard all by improving its performance (ii) SOLOv2 [40] as an anchor-free SOTA method.

YOLACT: Following YOLACT [1], we train and test RS-YOLACT by images with size 550×550 for 55 epochs. Instead of searching for epochs to decay learning rate, carefully tuned for YOLACT as 20, 42, 49 and 52, we simply adopt cosine annealing with an initial learning rate of 0.006. Then, we remove (i) OHEM, (ii) semantic segmentation head, (iii) carefully tuned task weights (i.e. $\lambda_{box} = 1.5$, $\lambda_{mask} = 6.125$) and (iv) size-based normalization (i.e. normalization of mask head loss of each instance by the ground-truth area). Removing each heuristic ensues a slight to significant performance drop (at least requires retuning of λ_t – Table 6). After these simplifications, our RS-YOLACT improves baseline by 1.5 mask AP and 3.3 box AP.

SOLOv2: Following Wang et al. [40], we train anchor-free SOLOv2 with RS Loss for 36 epochs using multiscale training on its two different settings: (i) SOLOv2-light is the real-time setting with ResNet-34 and images with size 448×448 at inference. We use 32 images/batch and learning rate 0.012 for training. (ii) SOLOv2 is the SOTA setting with ResNet-101 and images with size 1333×800 at inference. We use 16 images/batch and learning rate 0.006 for training. Since SOLOv2 does not have a box regression head, we use Dice coefficient as the continuous labels of RS Loss (see Supp.Mat. for an analysis of using different localisation qualities as labels for instance segmentation). Again, RS Loss performs better than the baseline (i.e. Focal Loss and Dice Loss) only by tuning the learning rate (Table 7).

6.2.3 Comparison with SOTA

We use our RS-Mask R-CNN (i.e. standard Mask R-CNN with RS Loss) to compare with SOTA methods. In order to fit in 16GB memory of our V100 GPUs and keep all settings unchanged, we limit the number of maximum proposals in the mask head by 200, which can simply be omitted for GPUs with larger memory. Following our counterparts [40, 41], we first train RS-Mask R-CNN for 36 epochs with multiscale training between $[640, 800]$ using ResNet-101 and reach 40.6 mask AP (Table 8), improving Mask R-CNN by 2.3 mask AP and outperforming all SOTA methods by a notable margin (~ 1 AP). Then, we train RS-Mask R-CNN+ (i.e. standard Mask R-CNN except upsampling of FPN is lightweight Carafe [39]) also by extending the multiscale range to $[480, 960]$ and reach 42.0 mask AP, which even outperforms all models with DCN. With DCN [45] on ResNet-101, our RS-Mask R-CNN+ reaches 43.9 mask AP.

Table 6. RS-YOLACT does not employ any additional training heuristics and outperforms YOLACT by significant margin.

Method	Additional Training Heuristics			Segmentation Performance				Detection Performance				H# ↓
	OHEM [36]	Size-based Norm.	Sem.Segm. Head	AP ↑	AP_{50} ↑	AP_{75} ↑	oLRP ↓	AP ↑	AP_{50} ↑	AP_{75} ↑	oLRP ↓	
YOLACT [1]	✓	✓	✓	28.4	47.7	29.1	75.4	30.5	52.3	31.8	73.9	5
		✓	✓	15.1	27.1	15.0	86.6	12.6	27.8	10.0	88.5	4
	✓		✓	21.7	40.2	20.7	80.5	30.4	52.2	31.4	74.1	5
	✓	✓		28.1	47.5	28.7	75.6	30.5	51.9	32.0	74.0	4
				13.6	26.4	12.4	87.5	15.1	32.9	12.1	86.3	3
RS-YOLACT				**29.9**	**50.5**	**30.6**	**74.7**	**33.8**	**54.2**	**35.4**	**71.8**	**1**

Table 7. Comparison on anchor-free SOLOv2.

Method	Backbone	AP	AP_{50}	AP_{75}	oLRP ↓	H# ↓
SOLOv2-light	ResNet-34	32.0	50.7	33.7	73.5	3
RS-SOLOv2-light	ResNet-34	**32.6**	**51.7**	**34.2**	**72.7**	**1**
SOLOv2	ResNet-101	39.1	59.8	41.9	67.3	3
RS-SOLOv2	ResNet-101	**39.7**	**60.6**	**42.2**	**66.9**	**1**

Table 8. Comparison with SOTA for instance segmentation on COCO *test-dev*. All methods (except *) use ResNet-101. The result of the similarly trained Mask R-CNN is acquired from Chen et al. [9]. +: upsampling of FPN is Carafe, *:ResNeXt-64x4d-101

	Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
w/o DCN	Polar Mask [42]	32.1	53.7	33.1	14.7	33.8	45.3
	Mask R-CNN [9]	38.3	61.2	40.8	18.2	40.6	54.1
	SOLOv2 [40]	39.7	60.7	42.9	17.3	42.9	**57.4**
	Center Mask [41]	39.8	–	–	21.7	42.5	52.0
	BCNet [16]	39.8	61.5	43.1	22.7	42.4	51.1
	RS-Mask R-CNN (Ours)	40.6	62.8	43.9	22.8	43.6	52.8
	RS-Mask R-CNN+ (Ours)	**42.0**	**64.8**	**45.6**	**24.2**	**45.1**	54.6
w DCN	Mask-scoring R-CNN [14]	39.6	60.7	43.1	18.8	41.5	56.2
	BlendMask [4]	41.3	63.1	44.6	22.7	44.1	54.5
	SOLOv2 [40]	41.7	63.2	45.1	18.0	45.0	**61.6**
	RS-Mask R-CNN+ (Ours)	**43.9**	**67.1**	**47.6**	**25.6**	**47.0**	57.8
	RS-Mask R-CNN+* (Ours)	44.8	68.4	48.6	27.1	47.9	58.3

Table 9. Contribution of the components of RS Loss on ATSS.

Architecture	RS Loss	score-based w.	task bal.	AP	H# ↓
ATSS+ResNet50 w.o. aux. head				38.7	3
	✓			39.7	2
	✓	✓		39.8	2
	✓	✓	✓	39.9	1

Table 10. Ablation with different degrees of imbalance on different datasets and samplers. Number of negatives (neg) corresponding to a single positive (pos) averaged over the iterations of the first epoch is presented. Quantitatively, pos:neg ratio varies between 1:7 to 1:10470. RS Loss successfully trains different degrees of imbalance without tuning (c.f. Tables 1 and 5). Details: Supp.Mat.

Dataset	Sampler		Desired Neg #		Actual Neg #		AP
	RPN	R-CNN	RPN	R-CNN	RPN	R-CNN	
COCO	Random	Random	1	3	7	702	38.5
	None	Random	1	N/A	6676	702	39.3
	None	None	N/A	N/A	6676	1142	39.6
LVIS	None	None	N/A	N/A	3487	10470	25.2

6.3. Ablation Experiments

Contribution of the components: Replacing Focal Loss by RS Loss improves the performance significantly (1 AP - Table 9). Score-based weighting has a minor contribution and value-based task balancing simplifies tuning.

Robustness to imbalance: Without tuning, RS Loss can train models with very different imbalance levels successfully (Table 10): Our RS Loss (i) yields 38.5 AP on COCO with the standard random samplers (i.e. data is relatively balanced especially for RPN), (ii) utilizes more data when the samplers are removed, resulting in ~ 1 AP gain (38.5 to 39.6 AP), and (iii) outperforms all counterparts on the long-tailed LVIS dataset (c.f. Table 5), where the imbalance is extreme for R-CNN (pos:neg ratio is $1 : 10470$ - Table 10). Section S3.5 in Supp.Mat. presents detailed discussion.

Contribution of the sorting error: To see the contribution of our additional sorting error, during training, we track Spearman's ranking correlation coefficient (ρ) between IoUs and classification scores, as an indicator of the sorting quality, with and without our additional sorting error (see Eq. 6-8). As hypothesized, using sorting error improves sorting quality, ρ, averaged over all/last 100 iterations, from $0.38/0.42$ to $0.42/0.47$ for RS-R-CNN.

Effect on Efficiency: On average, one training iteration of RS Loss takes around $1.5\times$ longer than score-based losses. See Section S3.6 in Supp.Mat. for more discussion on the effect of RS Loss on training and inference time.

7. Conclusion

In this paper, we proposed RS Loss as a ranking-based loss function to train object detectors and instance segmentation methods. Unlike existing ranking-based losses, which aim to rank positives above negatives, our RS Loss also sorts positives wrt. their localisation qualities, which is consistent with NMS and the performance measure, AP. With RS Loss, we employed a simple, loss-value-based, tuning-free heuristic to balance all heads in the visual detectors. As a result, we showed on seven diverse visual detectors that RS Loss both consistently improves performance and significantly simplifies the training pipeline.

Acknowledgments: This work was supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) (under grants 117E054 and 120E494). We also gratefully acknowledge the computational resources kindly provided by TÜBİTAK ULAKBIM High Performance and Grid Computing Center (TRUBA) and Roketsan Missiles Inc. used for this research. Dr. Oksuz is supported by the TÜBİTAK 2211-A Scholarship. Dr. Kalkan is supported by the BAGEP Award of the Science Academy, Turkey.

References

[1] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 2, 5, 7, 8

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: Delving into high quality object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 5, 6

[3] Jiale Cao, Hisham Cholakkal, Rao Muhammad Anwer, Fahad Shahbaz Khan, Yanwei Pang, and Ling Shao. D2det: Towards high quality object detection and instance segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 7

[4] Hao Chen, Kunyang Sun, Zhi Tian, Chunhua Shen, Yongming Huang, and Youliang Yan. Blendmask: Top-down meets bottom-up for instance segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 8

[5] Joya Chen, Dong Liu, Tong Xu, Shilong Zhang, Shiwei Wu, Bin Luo, Xuezheng Peng, and Enhong Chen. Is sampling heuristics necessary in training deep object detectors? *arXiv*, 1909.04868, 2019. 2, 5

[6] Kean Chen, Weiyao Lin, Jianguo li, John See, Ji Wang, and Junni Zou. Ap-loss for accurate one-stage object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, pages 1–1, 2020. 1, 2, 3, 4, 6, 7

[7] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. Hybrid task cascade for instance segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[8] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv*, 1906.07155, 2019. 4, 5

[9] Xinlei Chen, Ross Girshick, Kaiming He, and Piotr Dollár. Tensormask: A foundation for dense object segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 8

[10] Yihong Chen, Zheng Zhang, Yue Cao, Liwei Wang, Stephen Lin, and Han Hu. Reppoints v2: Verification meets regression for object detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 7

[11] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 7

[12] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask R-CNN. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 5, 6, 7

[13] Yihui He, Chenchen Zhu, Jianren Wang, Marios Savvides, and Xiangyu Zhang. Bounding box regression with uncertainty for accurate object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[14] Zhaojin Huang, Lichao Huang, Yongchao Gong, Chang Huang, and Xinggang Wang. Mask scoring r-cnn. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 5, 6, 7, 8

[15] Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. Acquisition of localization confidence for accurate object detection. In *The European Conference on Computer Vision (ECCV)*, 2018. 2, 5, 6

[16] Lei Ke, Yu-Wing Tai, and Chi-Keung Tang. Deep occlusion-aware instance segmentation with overlapping bilayers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 8

[17] Kang Kim and Hee Seok Lee. Probabilistic anchor assignment with iou prediction for object detection. In *The European Conference on Computer Vision (ECCV)*, 2020. 2, 4, 5, 6, 7

[18] Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2, 4, 5, 6, 7

[19] Yanghao Li, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Scale-aware trident networks for object detection. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 6, 7

[20] Yu Li, Tao Wang, Bingyi Kang, Sheng Tang, Chunfeng Wang, Jintao Li, and Jiashi Feng. Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 7

[21] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4, 5, 6

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 42(2):318–327, 2020. 1, 4, 7

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *The European Conference on Computer Vision (ECCV)*, 2014. 4

[24] Peidong Liu, Gengwei Zhang, Bochao Wang, Hang Xu, Xiaodan Liang, Yong Jiang, and Zhenguo Li. Loss function discovery for object detection via convergence-simulation driven search. In *International Conference on Learning Representations (ICLR)*, 2021. 6

[25] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. In *The European Conference on Computer Vision (ECCV)*, 2016. 6

[26] Kemal Oksuz, Baris Can Cam, Emre Akbas, and Sinan Kalkan. Localization recall precision (LRP): A new performance metric for object detection. In *The European Confer-*

ence on Computer Vision (ECCV), 2018. 2, 6

[27] Kemal Oksuz, Baris Can Cam, Emre Akbas, and Sinan Kalkan. A ranking-based, balanced loss function unifying classification and localisation in object detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1, 2, 3, 5, 6, 7

[28] Kemal Oksuz, Baris Can Cam, Sinan Kalkan, and Emre Akbas. Imbalance problems in object detection: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, pages 1–1, 2020. 1, 2

[29] Kemal Oksuz, Baris Can Cam, Sinan Kalkan, and Emre Akbas. One metric to measure them all: Localisation recall precision (lrp) for evaluating visual detection tasks. *arXiv*, 2011.10772, 2020. 6

[30] Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin. Libra R-CNN: Towards balanced learning for object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 6

[31] Qi Qian, Lei Chen, Hao Li, and Rong Jin. Dr loss: Improving object detection by distributional ranking. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 39(6):1137–1149, 2017. 1, 2, 5

[33] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4, 6

[34] Michal Rolínek, Vít Musil, Anselm Paulus, Marin Vlastelica, Claudio Michaelis, and Georg Martius. Optimizing rank-based metrics with blackbox differentiation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[35] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958. 2, 3

[36] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 7, 8

[37] Jingru Tan, Xin Lu, Gang Zhang, Changqing Yin, and Quanquan Li. Equalization loss v2: A new gradient balance approach for long-tailed object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 7

[38] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 2

[39] Jiaqi Wang, Kai Chen, Rui Xu, Ziwei Liu, Chen Change Loy, and Dahua Lin. Carafe: Content-aware reassembly of features. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 5, 6, 7

[40] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic and fast instance segmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 5, 7, 8

[41] Yuqing Wang, Zhaoliang Xu, Hao Shen, Baoshan Cheng, and Lirong Yang. Centermask: Single shot instance segmentation with point representation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 7, 8

[42] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 8

[43] Hongkai Zhang, Hong Chang, Bingpeng Ma, Naiyan Wang, and Xilin Chen. Dynamic r-cnn: Towards high quality object detection via dynamic training. In *The European Conference on Computer Vision (ECCV)*, 2020. 6, 7

[44] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 4, 5, 6, 7

[45] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 6, 7