Score to Learn: A Comparative Analysis of Scoring Functions for Active Learning in Robotics

Riccardo Grigoletto(✉), Elisa Maiettini, and Lorenzo Natale

Humanoid Sensing and Perception, Istituto Italiano di Tecnologia, Genoa, Italy
ricgri@kth.se

Abstract. Accurately detecting objects in unconstrained settings is crucial for robotic agents, such as humanoids, that function in ever-changing environments. Current deep learning based methods achieve remarkable performance on this task on general purpose benchmarks and they are therefore appealing for robotics. However, their high accuracy comes at the price of computationally expensive off-line training and extensive human labeling. These aspects make their adoption in robotics challenging, since they prevent rapid model adaptation and re-training to novel tasks and conditions. Nonetheless, robots, and especially humanoids, being embodied in the surrounding environment, have access to streams of data from their sensors that, even though without supervision, might contain information of the objects of interest. The Weakly-supervised Learning (WSL) framework offers a set of tools to tackle these problems in general-purpose Computer Vision. In this work, we aim at investigating their adoption in the robotics domain which is still at a preliminary stage. We build on previous work, studying the impact of different, so called, scoring functions, which are at the core of WSL methods, on Pascal VOC, a general purpose dataset, and a prototypical robotic setting, i.e. the iCubWorld-Transformations dataset.

Keywords: Object detection · Active learning · Scoring function · Robotics

1 Introduction

Localizing and recognizing objects of interest is a crucial problem in modern robotic applications. Current approaches to address this task are based on Convolutional Neural Networks (CNN) [1], like, e.g., Mask R-CNN [2], EfficientDet [3] and YoloV4 [4]. These methods achieve remarkable performance on standard object detection benchmarks like Pascal VOC [5], Imagenet [6] and MS COCO [7]. However, they typically rely on Supervised Learning, therefore, they require carefully annotated training data to be optimized. For tasks like object detection or instance segmentation, the image annotation process is typically highly expensive as it requires an expert to manually provide both the names and locations (in terms of bounding box or contour, respectively) of all the objects of

M. Vincze et al. (Eds.): ICVS 2021, LNCS 12899, pp. 55–67, 2021.
https://doi.org/10.1007/978-3-030-87156-7_5

interest in the image. For this reason, these methods are not suited for agents that operate in unconstrained environments (like e.g., humanoids), which require the ability to quickly update the current model to novel conditions. It has been shown [8] that in constrained scenarios it is possible to acquire automatically annotated images, exploiting a human robot interaction and additional information from the other sensory modalities of a humanoid, like iCub [9]. However, recently it has been shown [8,10] that such an approach has limited generalization capabilities and that performance drop when the robot is asked to recognize objects in a different context.

Nonetheless, robots are autonomous agents that can actively explore the surrounding environment, having access to streams of images that, even if without supervision, might contain the objects of interest in different view poses and conditions. Therefore, they convey useful information for model adaptation or retraining, but they cannot be used within the Supervised Learning framework as they lack exact annotations. Moreover, they are typically redundant and strongly correlated in time. In these cases, Weakly-supervised Learning (WSL) [11,12] can be considered. This is a Machine Learning framework which targets those scenarios where it is required to learn from partially annotated data. For this work, the sub-classes of methods of WSL that are more relevant are Active Learning (AL) and Semi-supervised Learning (SSL). In particular, in AL [13,14], the informative unlabeled images are asked for annotations to an expert, with the aim of minimizing the labeling effort. The definition of the informativeness of an image is at the basis of each AL algorithm. SSL, instead, attempts to exploit the unlabeled images without querying for human annotation, by e.g., using high-confident predictions as pseudo ground-truth, in a self-supervised fashion. In both AL and SSL, it is fundamental to define evaluation functions which allow to express both the informativeness of the unlabeled images related to the task at hand and the confidence level of the predicted information. These functions are typically called *Scoring functions* [13,14].

Lately, WSL has been successfully applied to the object detection task [15–18], however their adoption in robotics is still at a preliminary stage. For instance, in [10], an on-line learning method for object detection [19] has been successfully integrated with a WSL pipeline [20], while in [21], different AL and SSL selection policies have been tested in robotics. The aim of this paper is, instead, to analyze the impact of different scoring functions, as they represent a core component of WSL methods, in a prototypical robotic scenario. Specifically, we compare different scoring functions, drawn from the Computer Vision literature, on two datasets for object detection: (i) the general purpose Pascal VOC [5] and (ii) the robotic dataset iCubWorld-Transformations [22]. Moreover, we provide insights on how the different functions affect the detection performance in terms of accuracy, training time and labeled data requirement on the two different tasks. We released the code to reproduce the experiments[1].

[1] https://github.com/RiccardoGrigoletto/SSM-Pytorch.

In the remaining of this paper, we report on the state-of-the-art on object detection and WSL (Sect. 2). Then, we describe the proposed method (Sect. 3) and we report on our experimental analysis (Sect. 4). Finally, we conclude commenting the obtained results (Sect. 5).

2 Related Work

In this section we present the state-of-the-art on object detection and WSL.

2.1 Deep Learning Based Object Detection

Approaches to the object detection task can be divided in two different categories: (i) grid-based and (ii) region-based detectors. In grid-based methods, for each image, classifiers are directly applied over a dense grid of cells, representing different object locations, scales, and aspect ratios. Recent examples of grid-based methods are: YOLO [4,23,24], SSD [25,26], RetinaNet [27], RefineDet [28] and CornerNet [29]. Instead, in region-based approaches, a previous step of region proposal generation is performed to predict a sparse set of candidate locations that might contain the objects of interest and that need to be further classified. As an example, Region-CNN (R-CNN) [30] can be mentioned, together with its optimizations: Fast R-CNN [31], Faster R-CNN [32], Region-FCN [33] and Mask R-CNN [2]. Typically, grid-based methods prove to be faster than region-based ones, but less precise [34].

All the aforementioned methods achieved high performance on general purpose Computer Vision benchmarks [5–7]. However, their application in robotics is not straightforward if fast adaptation capabilities are required. Indeed, these methods are generally composed of monolithic deep CNN-based architectures, trained end-to-end via stochastic gradient descent and back-propagation, thus requiring long training time and a large amount of carefully annotated images. Both these characteristics prevent fast and efficient adaptation to novel conditions. While the first issue has been recently addressed in [19,35–37], in this work we tackle the requirement of labeled data, by investigating WSL techniques, more specifically the scoring function component, to reduce the labeling human effort.

2.2 Weakly-Supervised Learning of Object Detection

The introduction of pipelines that allow to collect automatically annotated images (like, e.g. [8]) has alleviated the manually labeled data requirement of object detection methods. However, their usage may limit generalization capabilities of the learned model, since they typically require constrained scenarios for their functioning [8]. A solution to this problem is to consider WSL techniques, which allow to exploit unlabeled data to update and improve detection models [11,12]. AL and SSL (which have been introduced in Sect. 1) are two of the tools provided by the WSL framework. While their application to the object

classification problem is well known (see, e.g. [14,38,39]), their adaptation to the object detection task is not straightforward, since each image can contain more than one object and the scoring function needs to take all of them into account. Moreover, the information of the location of the objects has to be considered as well. Recent work has been done in this direction [15–17,40,41]. Moreover, lately, SSL and AL have also been combined in a unique pipeline, called Self-supervised Sample Mining (SSM) [20] that is composed of (i) a CNN-based object detection method and (ii) a scoring function called Cross Image Validation (ICV) [20]. This latter is used to evaluate with a score the predictions on each unlabeled image. The produced score is used by the model to decide whether to ask it for annotation (AL) or accept the proposed prediction as a training label for that image (SSL). Finally, the newly obtained training set is used to fine-tune the detector. This process is repeated for different iterations over the unlabeled dataset.

While WSL techniques, and specifically SSM, have been recently integrated [10,21] with an on-line learning method for object detection for robotics [19], their adoption in robotics is still at a preliminary stage. In this work, we aim at investigating the impact of different scoring functions, a core component in all WSL pipelines, in a robotic scenario. Specifically, we integrate main state-of-the-art scoring functions with the SSM pipeline and we evaluate their robustness and efficiency on both general purpose and robotic datasets.

3 Methods

In this work, we consider the scenario of a robot provided with an object detection model pre-trained on a labeled, but scarce, dataset. The robot has access to a second set of unlabeled images and it can use them to refine the given detection model. We tackle this scenario with the WSL framework. Our aim is that of carrying out a systematic experimental evaluation, investigating the impact of the scoring function component in a WSL pipeline for object detection for robotics. In this section, we present the pipeline that has been used for our experimental analysis (Sect. 3.1) and the considered scoring functions (Sect. 3.2).

3.1 Overview of the Pipeline

The proposed pipeline builds on the SSM [20]. It is composed of three main building blocks (refer to Fig. 1 for a pictorial representation): (i) the *Object Detection module*, (ii) the *Dataset* and (iii) the *WSL module*.

Object Detection Module. For this part, in our experiments we chose the state-of-the-art approach Faster-RCNN [32]. This is a region-based method (see Sect. 2.1) and it is composed of: (i) a CNN based feature extractor, which computes convolutional descriptors for each image, (ii) a Region Proposal Network (RPN), which predicts a set of rectangular candidate regions in the image that might contain the objects of interest and (iii) a final Detector which classifies and refines all these proposals, providing a final set of predicted detections.

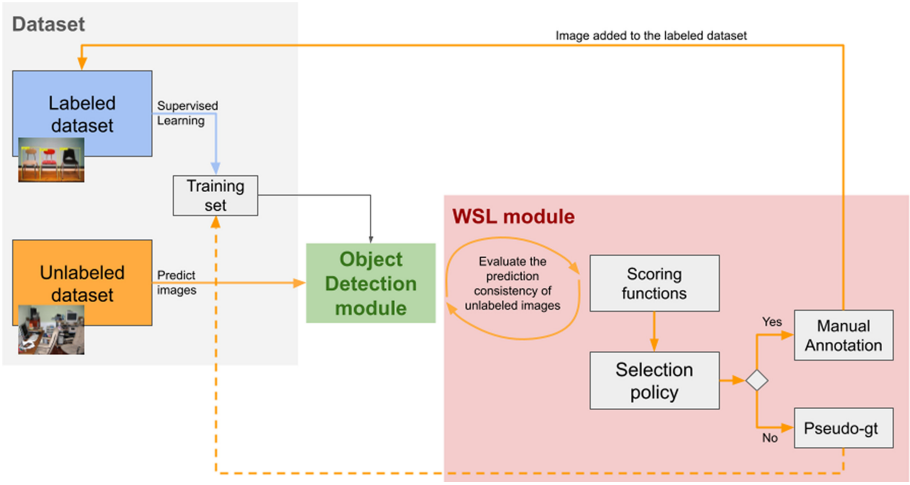

Fig. 1. Pictorial representation of the proposed pipeline. The blue arrows represent the *SL phase*, the orange ones represent the *WSL phase* (see Sect. 3.1 for details). (Color figure online)

In this work, for training Faster-RCNN we rely on the method proposed in [32]. Initially, we train it with the available labeled images, during the *Supervised Learning (SL) phase* (blue arrows in Fig. 1). Subsequently, the obtained detection model is iteratively refined using the unlabeled set of images, with the *WSL module*, during the *WSL phase* (orange arrows in Fig. 1).

Dataset. This component collects both the labeled and unlabeled sets of images at each iteration of the *WSL phase*. The former one is firstly used to pre-train the *Object Detection module* during the *SL phase*. Then, during the *WSL phase*, the unlabeled set is processed by the current *Object Detection module* and the predictions are evaluated by the *WSL module*. All the images with uncertain detections are asked for annotations (AL) and added to the labeled set, while all the confident ones are used as pseudo-groundtruth (SSL). Both of them are used as *Training set* for re-training the detection model during the current iteration of the *WSL phase*. At the end of each iteration, the *Training set* is re-initialized.

WSL Module. Finally, the *WSL module* consists of a (i) Scoring Function and a (ii) Selection Policy. During the *WSL phase*, the former one evaluates the predictions of the current *Object Detection module* on the unlabeled set of images, producing a consistency score [20] for each of them. The consistency score represents the confidence of the predictions and it is used by the Selection Policy to decide whether they are confident enough to be used as pseudo-labels (SSL) or if it is necessary to ask that image for manual annotation (AL). In this work, for the Selection Policy, we rely on the method proposed in [20]. Moreover, we refer to [21] for an empirical analysis of this latter component in a robotic

setting. Our main contribution is in the scoring function block. In the next section, we describe the ones that we considered for our experimental analysis.

3.2 Scoring Functions

A scoring function calculates a consistency score $S(x)$, given an image x from the unlabeled set of images I and the corresponding predictions from the current model. In our pipeline, the predictions of the *Object Detection module*, for each image $x \in I$, are represented by a set of bounding boxes B_x. For each $b \in B_x$, a vector of confidence scores K is predicted, of size n, where n is the number of classes of the considered task (we denote with C the set of classes). The j^{th} element in K represents the probability that the considered predicted box represents an instance of the j^{th} class. Typically, for each b, the predicted class c_1 corresponds to the index of the maximum value k_b^* in K. Therefore, k_b^* represents the probability that the bounding box b depicts an object of class c_1, i.e., $k_b^* = \max_{\{c_1 \in C\}}(\hat{p}(c_1|b))$. In this work, we consider five different scoring functions from the state-of-the-art of Computer Vision and we evaluate them in a robotic setting. Specifically, two of them (namely, Maximum Confidence and Margin Sampling) have been drawn from the image classification literature [13] and adapted as follows for object detection while the others (namely, Cross Image Validation, Localization Tightness and Localization Stability) have been proposed for general purpose object detection with the purpose of integrating them in a robotic pipeline. We describe each of them in the following paragraphs.

Maximum Confidence (MC) [13]. This function computes the consistency score of an image x as the average of the k_b^* values for all the boxes in B_x:

$$S(x) = \frac{1}{|B_x|} \sum_{b \in B_x} k_b^* \tag{1}$$

Margin Sampling (MS) [13]. This function compares the difference between the first and second maximum values in K for each $b \in B_x$. It is computed as follows:

$$S(x) = \frac{1}{|B_x|} \sum_{b \in B_x} M_1(b) \tag{2}$$

where $M_1(b)$ represents the score for the single bounding box $b \in B_x$, such that:

$$M_1(b) = |\max_{\{c_1 \in C\}}(\hat{p}(c_1|b)) - \max_{\{c_2 \in C \setminus c_1\}}(\hat{p}(c_2|b))| \tag{3}$$

Cross Image Validation (ICV) [20]. It measures the confidence of the detections for an image by examining each predicted box as follows: (i) each detection is pasted into L different annotated images and (ii) the current detection model is used to predict them. The new predictions are compared with the ones of the original image and the score function is defined as:

$$S(x) = \frac{1}{|B_x|} \sum_{b \in B_x} M_2(b) \tag{4}$$

$M_2(b)$ represents the score for the single bounding box $b \in B_x$ such that:

$$M_2(b) = \frac{1}{\sum\limits_{l \in B_L} \hat{p}(c_1|l)} \sum_{l \in B_L} \mathbf{1}(IoU(b,l) \geq \gamma)\hat{p}(c_1|l) \tag{5}$$

where B_L is the set of detections in the images in L corresponding to b. The $IoU(\cdot)$ is the Intersection over Union function, $\mathbf{1}(\cdot)$ is the indicator function and γ represents the acceptance threshold for an IoU ($\gamma = 0.5$, in our experiments).

Localization Tightness (LT) [40]. This function specifically applies to region-based object detection methods (see Sect. 2.1). It is computed as follows.

$$S(x) = \frac{1}{|B_x|} \sum_{b \in B_x} M_3(b) \tag{6}$$

where $M_3(b)$ represents the score for the single bounding box $b \in B_x$ such that:

$$M_3(b) = |IoU(r,b) + k_b^* - 1| \tag{7}$$

where r is the region candidate from which b originated. The intuition behind this scoring function is that if r and b are too different it means that the Detector heavily modified the candidate regions predicted by the RPN during the refinement (see Sect. 3.1). This represents a "disagreement" of the two models on the position and size of the bounding boxes in an image, therefore they would benefit from re-training with the correct labels for that image.

Localization Stability (LS) [40]. This function measures the confidence of a detection for an image by repeating the prediction step on noisy versions of the same image and examining the consistency of the detections. Specifically, if N different Gaussian noise levels are chosen, the current detection model is applied N times on the N differently corrupted images ($N = 5$, in our experiments). For each initial predicted bounding box b, the most overlapping bounding box b_n is associated, which has been predicted by the detection model on the n^{th} image of total N. The consistency score is computed as follows:

$$S(x) = \frac{\sum_{b \in B_x} k_b^* M_4(b)}{\sum_{b \in B_x} k_b^*} \tag{8}$$

where $M_4(b)$ represents the score for the single bounding box $b \in B_x$ such that:

$$M_4(b) = \frac{\sum_{n \in N} IoU(b, b_n)}{N} \tag{9}$$

4 Experiments

In this section, we present the experimental analysis carried out to evaluate and compare the scoring functions presented in Sect. 3.

4.1 Experimental Setup

For our analysis, we considered two different datasets: the Pascal VOC (VOC) [5] and the iCubWorld-Transformations (iCWT) [22]. Specifically, for VOC we used both train and validation sets of the two subsets, namely, *VOC2007* (∼5k images) and *VOC2012* (∼11k images), both depicting 20 object categories (which represent different animals, vehicles, furniture, etc.). The *VOC2007* is used for the *SL phase*, while *VOC2012* is used for the *WSL phase* (see Sect. 3.1). Therefore, in our experiments, *VOC2012* is treated as an unlabeled dataset. We used the test set of *VOC2007* to calculate accuracy (∼5k images). When using iCWT, instead, we selected 30 of the 200 available objects instances, gathering ∼2k, ∼6k and 4.5k images, respectively for the labeled, unlabeled training subsets and for the test set. This dataset has been acquired as described in [22], with a natural interaction with the iCub humanoid robot [9], simulating a teacher-learner scenario. The 200 depicted objects can be typically found in a domestic environment and, for each of them, several image sequences are available. For the acquisition procedure and the depicted objects, iCWT represents a suitable test bench to validate our system in the target robotic scenario. In the reported experiments, we chose ResNet50 [42] as CNN backbone for feature extraction for Faster R-CNN. In both cases, the training is done by fine-tuning a set of weights that has been pre-trained on MS COCO [7]. For the *SL phase*, we fine-tuned the network for 70k and 8k iterations for respectively VOC and iCWT while for the *WSL phase*, we iterate the selection policy for four times over the unlabeled part of the dataset, fine-tuning the weights, each time, for 20k and 4k iterations for respectively VOC and iCWT.

The evaluation is performed comparing 3 different metrics:

1 The mean Average Precision (mAP) as defined in [5].
2 The computational time[2] during the scoring function computation reported in terms of processed images per second (im/s). This aspect is critical in the considered robotics application.
3 The ratio between the number of images selected for manual annotation (AL) and the number of those that are automatically annotated with a self-supervision (SSL) (referred to as AL/SS ratio). This metric has practical relevance because it allows to understand how much the self-supervision is used by the different scoring functions.

We repeat each experiment for five trials for VOC and for three trials for iCWT and we present the results, reporting the mean and the standard deviation of the obtained results.

[2] The models have been trained on a single GPU Nvidia TESLA K40 and Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10 GHz.

4.2 Results Analysis on Pascal VOC

We report the obtained results in Fig. 2. In particular, we show the mAP and AL/SS ratio trends for growing numbers of annotated images, respectively, in Fig. 2A and Fig. 2B. As it can be noticed in Fig. 2A, the different scoring functions have similar mAP trends. This means that, for VOC, both approaches drawn from the image classification literature (MC and MS) and the ones based on the consistency of the predicted bounding boxes (ICV, LT and LS) present similar accuracy performance for growing numbers of annotations. Notably, however, MC turned out to be the one that leads to less accurate results for small annotation budgets and to a higher variability of the obtained mAP on the different experiment trials. Moreover, as it can be observed from Fig. 2B, ICV and LS present AL/SS ratios which are, respectively, ∼10 and ∼7 times higher than the other methods. On the contrary, MS, MC and LT present very low values. This means that, for instance, ICV and LS achieve roughly the same accuracy as LS and LT, with the same number of AL but less SSL.

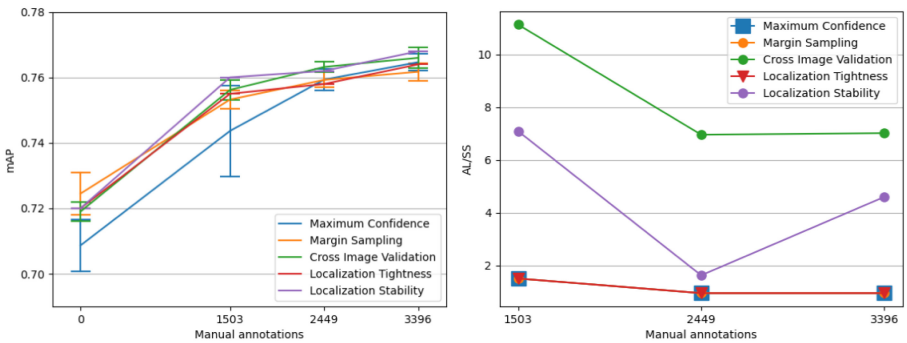

Fig. 2. Comparison of scoring functions on VOC in terms of mAP trend (A) and AL/SS ratio (B) for growing numbers of manual annotations.

Table 1. Time performance comparison for the different scoring functions on both datasets, VOC and iCWT.

Method	(im/s) on VOC	(im/s) on iCWT
MC	**2.19**	**2.07**
MS	**2.20**	**2.06**
ICV	0.56	1.06
LT	1.06	1.60
LS	0.55	1.04

4.3 Results Analysis on iCWT

In this section, we compare the scoring functions presented in Sect. 3.2 on the target robotic scenario, represented by the iCWT dataset. We report the obtained results in Fig. 3. Specifically, we show the mAP and AL/SS ratio trends for growing numbers of annotated images, respectively, in Fig. 3A and Fig. 3B. As it can be observed in Fig. 3A, mAP trends for iCWT for ICV and LS present the lowest slopes, while MC, LT and MS have the steepest ones especially for lower numbers of manual annotations. Notably, MC reaches the highest value of mAP (∼0.71) with only 606 annotated images.

As a comparison, we trained Faster R-CNN with all the available annotated images (i.e. ∼16k) in iCWT for the chosen task. The obtained model represents the upper-bound of the results presented in Fig. 3 since it uses the full dataset for training, achieving an mAP of 0.866. Even if the results in Fig. 3 are reasonably lower than the upper-bound, it is worth noticing that, for instance, with MS it has been possible to obtain an mAP of ∼0.71 with a significant lower amount of manually annotated images (606). This makes the proposed method a better trade-off than the training with the fully annotated dataset. Moreover, as it can be noted in Fig. 3B, as for VOC, ICV and LS present the highest AL/SS ratios. For instance, ICV (blue line in Fig. 3B) achieves 120 on the last step, meaning that for each image chosen for SSL, 120 are chosen for AL. However, differently from the VOC case, ICV and LS have the worst accuracy levels for early WSL iterations. This means that the samples chosen by the model as self-supervision for LT, MS and especially MC, significantly improved the overall detection accuracy.

Finally, Table 1 shows the time performance comparison. Specifically, the second column reports results for VOC, while the third one for iCWT. As it can be noted, in both cases MS and MC take considerably less time than all the other methods, while ICV and LS are the slowest methods. This is due to the

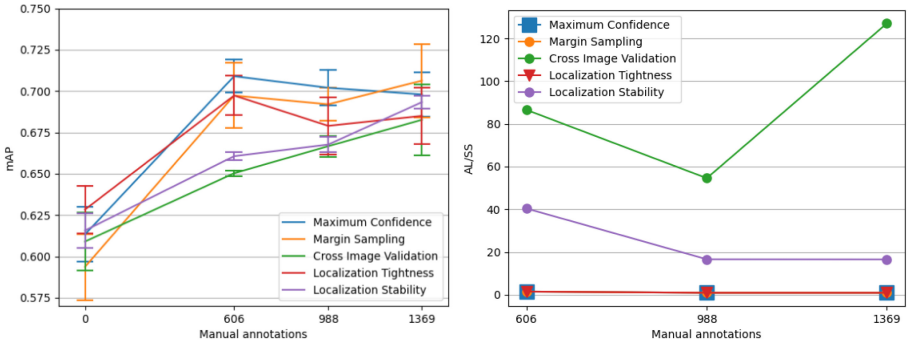

Fig. 3. Comparison of scoring functions on iCWT dataset in terms of mAP trend (A) and AL/SS ratio (B) for growing numbers of manual annotations. (Color figure online)

fact that both ICV and LS require to perform several inferences of Faster R-CNN for different images to evaluate the prediction consistency, while the others do it only once for the initial unlabeled image.

5 Discussion

In this work, we considered the scenario of a robot that is required to refine an object detection model with an incoming set of unlabeled images. We tackled this scenario with the WSL framework and we empirically evaluated the impact of the scoring function component in a WSL pipeline for object detection for robotics. Specifically, we compared five different scoring functions on both general purpose and robotics datasets, by means of the two benchmarks Pascal VOC and iCWT. Interestingly, we found out that while for Pascal VOC, the five methods have comparable accuracy performance, for the target robotic scenario they perform differently. Moreover, with the comparative analysis in terms of annotations and computation time required, we identified the most efficient methods. Notably, the three fastest scoring functions (namely, MC, MS and LT) present the best trends in terms of mAP and make a better use of self-supervision, representing valid options for a WSL based robotic application. We believe that the presented analysis provides useful insights on how to apply WSL techniques in a robotic setting, going towards the design of more efficient learning based robotic vision systems.

References

1. LeCun, Y., Bengio, Y., et al.: Convolutional networks for images, speech, and time series. In: The Handbook of Brain Theory and Neural Networks, vol. 3361.10, p. 1995 (1995)
2. He, K., et al.: Mask R-CNN. In: The IEEE International Conference on Computer Vision (ICCV), October 2017
3. Tan, M., Pang, R., Le, Q.V.: EfficientDet: scalable and efficient object detection (2020). arXiv:1911.09070 [cs.CV]
4. Bochkovskiy, A., Wang, C.-Y., Mark Liao, H.-Y.: YOLO4j: optimal speed and accuracy of object detection (2020). arXiv:2004.10934 [cs.CV]
5. Everingham, M., et al.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010). ISSN 0920-5691, 1573-1405. https://doi.org/10.1007/s11263-009-0275-4. http://link.springer.eom/10.1007/s11263-009-0275-4
6. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Pereira, F., et al. (eds.) Advances in Neural Information Processing Systems, vol. 25, pp. pp. 1097–1105. Curran Associates Inc. (2012). http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
7. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. arXiv:1405.0312 [cs], 20 February 2015. arXiv:1405.0312. URL: http://arxiv.org/abs/1405.0312. Accessed 21 May 2020

8. Maiettini, E., et al.: Interactive data collection for deep learning object detectors on humanoid robots. In: 2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids), pp. 862–868, November 2017. https://doi.org/10.1109/HUMAN0IDS.2017.8246973
9. Metta, G., et al.: The iCub humanoid robot: an open-systems platform for research in cognitive development. Neural Netw. Official J. Int. Neural Netw. Soc. **23**(8–9), 1125–34 (2010). https://doi.org/10.1016/j.neunet.2010.08.010. Jan
10. Maiettini, E., et al.: A weakly supervised strategy for learning object detection on a humanoid robot. In: 2019 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids), p. 8 (2019)
11. Zhou, Z.-H.: A brief introduction to weakly supervised learning. Nat. Sci. Rev. **5**(1), 44–53 (2018). https://academic.oup.com/nsr/article/5/1/44/4093912. https://doi.org/10.1093/nsr/nwxl06. ISSN 2095–5138, 2053–714X. Accessed 28 May 2020
12. Zhang, D., et al.: Weakly supervised object localization and detection: a survey. IEEE Trans. Pattern Anal. Mach. Intell. 1 (2021). https://doi.org/10.1109/TPAMI.2021.3074313
13. Settles, B.: Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences (2009)
14. Settles, B.: Active learning. In: Synthesis Lectures on Artificial Intelligence and Machine Learning (2012)
15. Aghdam, H.H., et al.: Active learning for deep detection neural networks. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). Seoul, Korea (South), pp. 3671–3679, IEEE, October 2019, ISBN 978-1-72814-803-8. https://ieeexplore.ieee.org/document/9009535/. https://doi.org/10.1109/ICCV.2019.00377. Accessed 16 June 2020
16. Haussmann, E., et al.: Scalable active learning for object detection. In: IEEE Intelligent Vehicles Symposium (IV), IEEE 2020, pp. 1430–1435 (2020)
17. Li, Y., Huang, D., Qin, D., Wang, L., Gong, B.: Improving object detection with *selective* self-supervised self-training. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12374, pp. 589–607. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58526-6_35
18. Dwibedi, D., Misra, I., Hebert, M.: Cut, paste and learn: surprisingly easy synthesis for instance detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1301–1310 (2017)
19. Maiettini, E., et al.: On-line object detection: a robotics challenge. Auton. Robot. 1573–7527 (2019). ISSN 0929-5593. http://link.springer.com/10.1007/S10514-019-09894-9. https://doi.org/10.1007/S10514-019-09894-9. Accessed 10 Feb 2020
20. Wang, K., et al.: Towards human-machine cooperation: self-supervised sample mining for object detection. arXiv:1803.09867 [cs], May 2018. http://eirxiv.org/abs/1803.09867. Accessed 30 Jan 2020
21. Maiettini, E., et al.: Data-efficient weakly-supervised learning for online object detection under domain shift in robotics (2020). arXiv:2012.14345
22. Pasquale, G., et al.: Are we done with object recognition? The iCub robot's perspective. Robot. Auton. Syst. 112, 260–281 (2019). ISSN: 09218890. arXiv:1709.09882. https://doi.org/10.1016/j.robot.2018.11.001. Accessed 13 Jan 2020
23. Redmon, J., et al.: You only look once: unified, real-time object detection. arXiv:1506.02640 [cs], 9 May 2016. arXiv:1506.02640. Accessed 26 May 2020
24. Redmon, J., Farhadi, A.: YOLOv3: an incremental improvement. CoRR abs/1804.02767 (2018). arXiv:1804.02767

25. Liu, W., et al.: SSD: single shot multibox detector, pp. 21–37. arXiv:1512.02325 [cs] 9905 (2016). arXiv:1512.02325. Accessed 26 May 2020. https://doi.org/10.1007/978-3-319-46448-0_2
26. Zhai, S., et al.: DF-SSD: an improved SSD object detection algorithm based on DenseNet and feature fusion. IEEE Access **8**, 24344–24357 (2020)
27. Lin, T.-Y., et al.: Focal loss for dense object detection, 7 February 2018. arXiv:1708.02002. Accessed 26 May 2020
28. Zhang, S., et al.: Single-shot refinement neural network for object detection. arXiv:1711.06897 [cs], 3 January 2018. arXiv:1711.06897. Accessed 26 May 2020
29. Law, H., Deng, J.: CornerNet: detecting objects as paired keypoints. In: arXiv:1808.01244 [cs], 18 March 2019. arXiv:1808.01244. http://cirxiv.org/abs/1808.01244. Accessed 26 May 2020
30. Girshick, R.B., et al.: Rich feature hierarchies for accurate object detection and semantic segmentation. CoRR abs/1311.2524 (2013). arXiv:1311.2524
31. Girshick, R.: Fast R-CNN. arXiv:1504.08083 [cs], 27 September 2015. arXiv:1504.08083. Accessed 20 May 2020
32. Ren, S., et al.: Faster R-CNN: towards real-time object detection with region proposal networks. arXiv:1506.01497 [cs], January 2016. arXiv:1506.01497. Accessed 29 Jan 2020
33. Dai, J., et al.: R-FCN: object detection via region-based fully convolutional networks. arXiv:1605.06409 [cs], 21 June 2016. arXiv:1605.06409. Accessed 26 May 2020
34. Huang, J., et al.: Speed/accuracy trade-offs for modern convolutional object detectors. arXiv:1611.10012 [cs], 24 April 2017. arXiv:1611.10012. Accessed 28 May 2020
35. Maiettini, E., et al.: Speeding-up object detection training for robotics with FALKON. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), October 2018, pp. 5770–5776. https://doi.org/10.1109/IR0S.2018.8593990
36. Ceola, F., et al.: Fast region proposal learning for object detection for robotics (2020). arXiv:2011.12790 [cs.CV]
37. Ceola, F., et al.: Fast object segmentation learning with kernel-based methods for robotics (2020). arXiv:2011.12805 [cs.CV]
38. Kirsch, A., van Amersfoort, J., Gal, Y.: BatchBALD: efficient and diverse batch acquisition for deep Bayesian active learning. In: NeurIPS (2019)
39. Ash, J.T., et al.: Deep batch active learning by diverse, uncertain gradient lower bounds, January 2020. https://openreview.net/forum?id=0HjEAtQNNWD. Accessed 26 Oct 2020
40. Kao, C.-C., Lee, T.-Y., Sen, P., Liu, M.-Y.: Localization-aware active learning for object detection. In: Jawahar, C.V., Li, H., Mori, G., Schindler, K. (eds.) ACCV 2018. LNCS, vol. 11366, pp. 506–522. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-20876-9_32
41. Desai, S.V., et al.: An adaptive supervision framework for active learning in object detection. arXiv preprint arXiv:1908.02454 (2019)
42. He, K., et al.: Deep residual learning for image recognition. arXiv:1512.03385 [cs], December 2015. Accessed 09 July 2020