UNBIASED TEACHER FOR SEMI-SUPERVISED OBJECT DETECTION

Anonymous authors
Paper under double-blind review

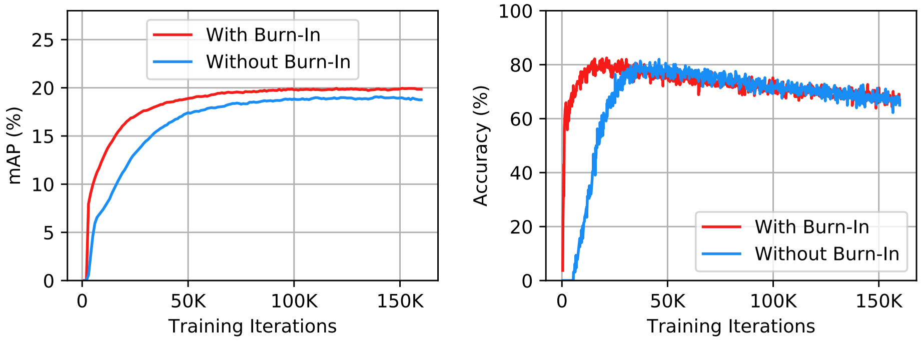
A APPENDIX

A.1 ADDITIONAL ABLATION STUDY

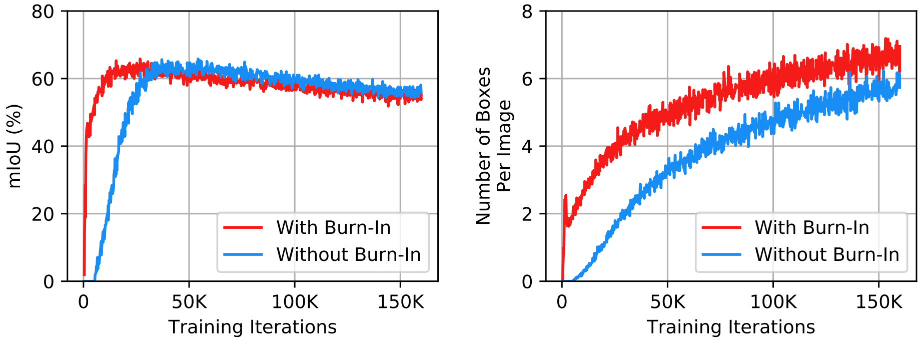
In addition to the ablation studies provided in the main paper, we further ablate Unbiased Teacher in the following sections.

A.1.1 EFFECT OF BURN-IN STAGE

It is crucial to have a good initialization for both *Student* and *Teacher* models. We thus present a comparison between the model with and without the Burn-In stage in Figure 1. We observe that, with the Burn-In stage, the model can derive more accurate pseudo-boxes in the early stage of the training. As a result, the model can achieve higher accuracy in the early stage of the training, and it also achieves better results when the model is converged.



(a) mAP (b) Box Accuracy



(c) mIoU (d) Number of Pseudo-Boxes

Figure 1: In the case of *COCO-standard* 1% labeled data, (a) Unbiased Teacher with Burn-In stage achieve higher mAP against Unbiased Teacher without Burn-In stage. Using Burn-In Stage results in the early improvement of (b) box accuracy and (c) mIoU. (d) Unbiased Teacher with Burn-In stage can derive more pseudo-boxes than Unbiased Teacher without Burn-In stage.

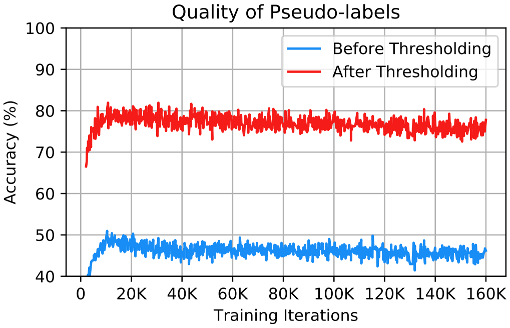
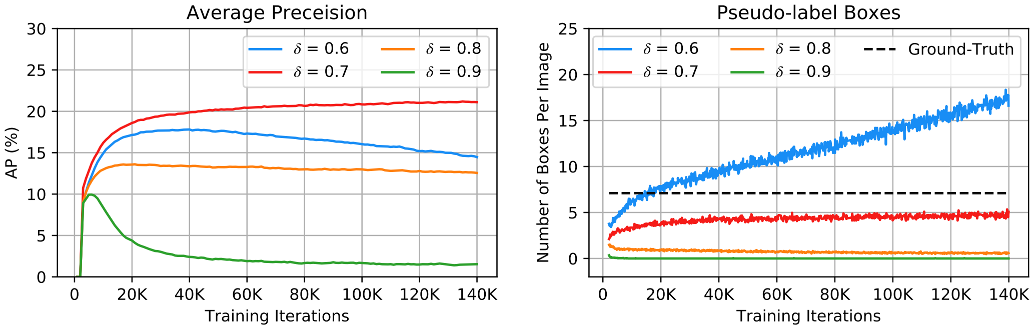


Figure 2: Pseudo-label accuracy improvement with the use of confidence thresholding. We measure the accuracy by comparing the ground-truth labels and predicted labels before and after confidence thresholding. This result indicates that confidence thresholding can significantly improve the quality of pseudo-labels.



(a) (b)

Figure 3: (a) Validation AP and (b) number of pseudo-label bounding boxes per image with various pseudo-labeling thresholds δ. With an excessively low threshold (*e.g.,* $\delta = 0.6$), the model has a lower AP, as it predicts more pseudo-labeled bounding boxes compared to the number of bounding boxes in ground-truth labels. On the other hand, the performance of the model using an excessively high threshold (*e.g.,* $\delta = 0.9$) drops as it cannot predict sufficient number of bounding boxes in its generated pseudo-labels.

A.1.2 EFFECT OF PSEUDO-LABELING THRESHOLD

We apply confidence thresholding to filter these low-confidence predicted bounding boxes, which are more likely to be false-positive instances. To show the effectiveness of thresholding, we first provide the accuracy of predicted bounding boxes before and after the pseudo-labeling in Figure 2.

When varying the threshold value δ from 0 to 0.9, as expected, the number of generated pseudo-boxes increases as the threshold δ reduces (Figure 3). The model using excessively high threshold (*e.g.,* $\delta = 0.9$) cannot perform satisfactory results, as the number of generated pseudo-labels is very low. On the other hand, the model using a low threshold (*e.g.,* $\delta = 0.6$) also cannot achieve favorable results since the model generates too many bounding boxes, which are likely to be false-positive instances. We also observe that the model cannot even converge if the threshold is below 0.5.

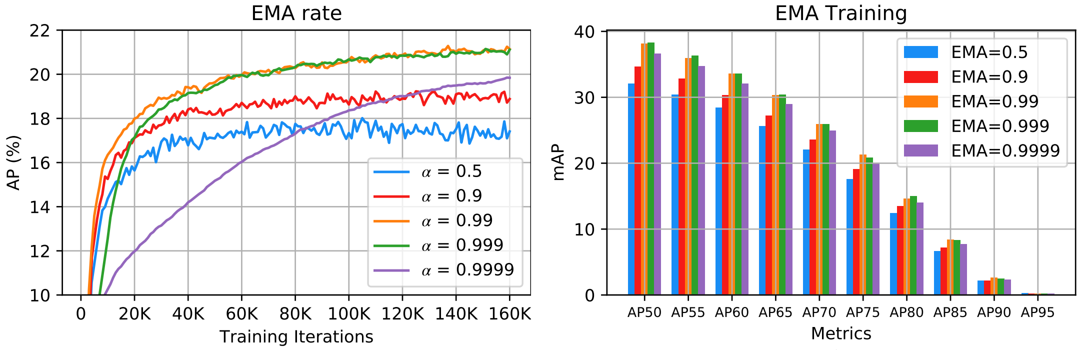
A.1.3 EFFECT OF EMA RATES

We also evaluate the model using various EMA rate α from 0.5 to 0.9999 and present the mAP result of the Teacher model in Figure 4. We observe that, with a smaller EMA rate (*e.g.,* $\alpha = 0.5$), the model has lower mAP and higher variance, as the Student contributes more to the Teacher model for each iteration. This implies the Teacher model is likely to suffer from the detrimental effect caused by noisy pseudo-labels. This unstable learning curve can be stabilized and improved as the EMA rate α increases. When the EMA rate α achieves 0.99, it performs the best mAP. However, if the

Table 1: Ablation study of varying unsupervised loss weight λ_u on the model trained using 10% labeled and 90% unlabeled data.

λ_u	1.0	2.0	4.0	5.0	6.0	8.0
AP (%)	29.30	30.64	31.82	32.00	31.80	Cannot Converge

EMA rate α keeps increasing, the teacher model will grow overly slow as the Teacher model derive the next model weight mostly from the previous Teacher model weight.



(a) (b)

Figure 4: Validation AP on the Teacher model with various MMA rates α. (a) With a small MMA rate (*e.g.,* $\alpha = 0.5$), the Teacher model has lower AP and larger variance. In contrast, as the MMA rate grows to 0.99, the Teacher model can gradually improve along the training iterations. However, when the MMA grows to 0.9999, the Teacher model grows overly slow but has lowest variance. (b) We breakdown the AP metric into APs from AP_{50} to AP_{95}.

A.1.4 EFFECT OF UNSUPERVISED LOSS WEIGHTS

To examine the effect unsupervised loss weights, we vary the unsupervised loss weight λ_u from 1.0 to 8.0 in the case of *COCO-standard* 10% labeled data. As shown in Table 1, with a lower unsupervised loss weight $\lambda_u = 1.0$, the model performs 29.30%. On the other hand, we observe that the model performs the best with unsupervised loss weight $\lambda = 5.0$. However, when the weight increases to 8.0, the training of the model cannot converge.

A.2 AP BREAKDOWN FOR COCO-STANDARD

We present an AP breakdown for COCO-standard 0.5% labeled data. Our proposed model can perform favorably against both STAC (Sohn et al., 2020) and CSD (Jeong et al., 2019). This trend appears in all evaluation metrics from AP_{50} to AP_{95}, as shown in Figure 5, and it confirms that our model is preferable for handling extremely low-label scenario compared to the state of the arts.

A.3 IMPLEMENTATION AND TRAINING DETAILS

Network and framework. Our implementation builds upon the Detectron2 framework (Wu et al., 2019). For a fair comparison, we follow the prior work (Sohn et al., 2020) to use Faster-RCNN with FPN (Lin et al., 2017) and ResNet-50 backbone (He et al., 2016) as our object detection network.

Training. At the beginning of the Burn-In stage, the feature backbone network weights are initialized by the ImageNet-pretrained model, which is same as existing works (Jeong et al., 2019; Tang et al., 2020; Sohn et al., 2020). We use the SGD optimizer with a momentum rate 0.9 and a learning rate 0.01, and we use constant learning rate scheduler. The batch size of supervised and unsupervised data are both 32 images. For the *COCO-standard*, we train 180k iterations, which includes $1/2/6/12/20$k iterations for $0.5\%/1\%/2\%/5\%/10\%$ in the Burn-In stage and the remaining iterations in the Teacher-Student Mutual Learning stage. For the *COCO-additional*, we train 360k

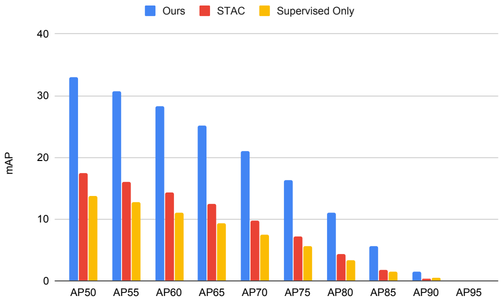


Figure 5: Evaluation metric breakdown of all methods on 0.5% labeled data.

iterations, which includes 90k iterations in the Burn-Up stage and the remaining 270k iterations in the Teacher-Student Mutual Learning stage.

Hyper-parameters. We use confidence threshold $\delta = 0.7$ to generate pseudo-labels for all our experiments, the unsupervised loss weight $\lambda_u = 4$ is applied for *COCO-standard* and *VOC*, and the unsupervised loss weight $\lambda_u = 2$ is applied for *COCO-additional*. We apply $\alpha = 0.9996$ as the EMA rate for all our experiments. Hyper-parameters used are summarized in Table 2.

Table 2: Meanings and values of the hyper-parameters used in experiments.

Hyper-parameter	Description	*COCO-standard* and VOC	*COCO-additional*
δ	Confidence threshold	0.7	0.7
λ_u	Unsupervised loss weight	4	2
α	EMA rate	0.9996	0.9996
b_l	Batch size for labeled data	32	16
b_u	Batch size for unlabeled data	32	16
γ	Learning rate	0.01	0.01

Data augmentation. As shown in Table 3, we apply randomly horizontal flip for weak augmentation and randomly add color jittering, grayscale, Gaussian blur, and cutout patches (DeVries & Taylor, 2017) for the strong augmentation. Note that we do not apply any image-level or box-level geometric augmentations, which are used in STAC (Sohn et al., 2020). In addition, we do not aggressively search the best hyper-parameters for data augmentations, and it is possible to obtain better hyper-parameters.

Evaluation Metrics. $AP_{50:95}$ is used to evaluate all methods following the prior works (Law & Deng, 2018; Sohn et al., 2020).

Table 3: Detail of data augmentations. Probability in the table indicates the probability of applying the corresponding image process.

Weak Augmentation

Process	Probability	Parameters	Descriptions
Horizontal Flip	0.5	-	None

Strong Augmentation

Process	Probability	Parameters	Descriptions
Color Jittering	0.8	(brightness, contrast, saturation, hue) = (0.4, 0.4, 0.4, 0.1)	Brightness factor is chosen uniformly from [0.6, 1.4], contrast factor is chosen uniformly from [0.6, 1.4], saturation factor is chosen uniformly from [0.6, 1.4], and hue value is chosen uniformly from [-0.1, 0.1].
Grayscale	0.2	None	None
GaussianBlur	0.5	(sigma_x, sigma_y) = (0.1, 2.0)	Gaussian filter with $\sigma_x = 0.1$ and $\sigma_y = 0.1$ is applied.
CutoutPattern1	0.7	scale=(0.05, 0.2), ratio=(0.3, 3.3)	Randomly selects a rectangle region in an image and erases its pixels. We refer the detail in Zhong et al. (2017).
CutoutPattern2	0.5	scale=(0.02, 0.2), ratio=(0.1, 6)	Randomly selects a rectangle region in an image and erases its pixels. We refer the detail in Zhong et al. (2017).
CutoutPattern3	0.3	scale=(0.02, 0.2), ratio=(0.05, 8)	Randomly selects a rectangle region in an image and erases its pixels. We refer the detail in Zhong et al. (2017).

REFERENCES

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

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 (CVPR)*, 2016.

Jisoo Jeong, Seungeui Lee, Jeesoo Kim, and Nojun Kwak. Consistency-based semi-supervised learning for object detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Kihyuk Sohn, Zizhao Zhang, Chun-Liang Li, Han Zhang, Chen-Yu Lee, and Tomas Pfister. A simple semi-supervised learning framework for object detection. *arXiv preprint arXiv:2005.04757*, 2020.

Peng Tang, Chetan Ramaiah, Ran Xu, and Caiming Xiong. Proposal learning for semi-supervised object detection. *arXiv preprint arXiv:2001.05086*, 2020.

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*, 2017.