Humble Teachers Teach Better Students for Semi Supervised Object Detection: Supplementary Materials

Yihe Tang Weifeng Chen Yijun Luo Yuting Zhang
Amazon Web Services
tangacademic@gmail.com {weifec,yijunl,yutingzh}@amazon.com

1. Weight of Unsupervised Loss

This section studies how tuning the weight of the unsupervised loss changes the model performance and documents our decision for some low-level details. Here we focus on the most important hyper-parameter defined in our model: the weight β of the unsupervised loss L_U.

All the experiments in this section are conducted on *MS-COCO train* with 10% labeled. We use the label-unlabeled data split 1 generated by code with the given random seed from [5]. We use Faster R-CNN [4] with FPN [3] and ResNet-50 [2] as our base model. It is worth noting that in this experiment, the number of region proposals used in the second stage of detection is 512, instead of 640 in the main paper and in Sec. 2.

The final loss L of our model is the sum of the supervised loss L_S and the unsupervised loss L_U with a scaling factor $\beta\frac{n_U}{n_S}$, as shown in Eqn. 1. n_U, n_S are the numbers of unlabeled and labeled images, and β is an additional weight on unsupervised loss. We use $\beta\frac{n_U}{n_S}$ to balance the unsupervised loss and the supervised loss in the model training.

$$L = L_S + \beta\frac{n_U}{n_S}L_U \tag{1}$$

Tab. 1 reports the detailed ablation results. We found that the performance deteriorates when β is too small or too large, indicating that a balance between supervised learning and unsupervised learning is crucial to good performance for our Humble Teacher. According to the ablation study, we set $\beta = 0.5$ across all experiments in the main paper and did not optimize them for particular experiments.

β	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
mAP	28.59	30.10	31.24	31.57	31.64	31.53	30.47	29.48

Table 1: The results of models with different unsupervised weight β on 10% labeled *MS-COCO 2017 train* (split 1), evaluated on the *MS-COCO 2017 val* set.

2. Ablation on Unsupervised Localization

We study whether the unsupervised loss on bounding box regression heads improves the performance of the final model. In this experiment, we compare enabling and disabling both bounding box regression heads for unsupervised loss. We keep other parameters the same. The basic experimental setup follows Sec. 1, except the number of region proposals used in the second stage of detection is set to 640, following the main paper. The results in Tab. 2 show that unsupervised learning on localization improves the final performance.

Model	with localization	without localization
mAP	31.83	30.78

Table 2: Comparison between models with unsupervised localization enabled and disabled. The models are trained on 10% labeled *MS-COCO 2017 train* (split 1), evaluated on the *MS-COCO 2017 val* set.

3. Hard Label Experiments

This section studies how different hyper-parameters impact the performance of models using hard labels. As we compare our models with the hard label models in the main paper, for fair comparison, we believe it is important to understand how hard label models perform the best.

There are two hyper-parameters for the hard label experiments: the weight β of pseudo-label loss, and the confidence threshold θ as the threshold for accepting hard pseudo-labels as training samples.

3.1. β: the Weight of Pseudo-Label Loss

We study how the weight of unsupervised loss β affects the model performance. Denote S_S as the labeled dataset and S_U as the unlabeled dataset. As shown in Eqn. 2, the total loss L of a hard label model is the sum of two losses: the loss on labeled images S_S and the loss on pseudo-labeled

images S_U. L_{rcnn} is the sum of standard Faster R-CNN losses. As in Sec. 1, n_U, n_S are numbers of the unlabeled images and labeled images, where β is an additional weight on unsupervised loss.

$$L = L_{\text{rcnn}}(x)|_{x \in S_S} + \beta \frac{n_U}{n_S} L_{\text{rcnn}}(y)|_{y \in S_U} \qquad (2)$$

For all the experiments, we perform experiments on *MS-COCO train* with 10% data labeled and use label-unlabeled data split 1 generated by code and random seed from [5]. We adopt Faster R-CNN [4] with FPN [3] and ResNet-50 [2] as our base model.

The results are shown in Fig. 1. We see that the model performs the best when β is between 0.09 and 0.10. For the overall best performance on different data splits, we used $\beta = 0.1$ for the hard label experiments in our main paper. We find that the soft-label model outperforms the best hard-label model even with extensive parameter tuning for the hard label model directly on *MS-COCO 2017 val*.

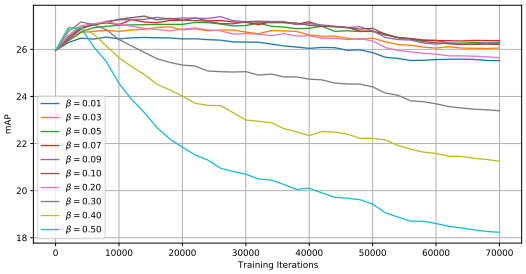

Figure 1: Comparison between hard label models with different β trained on 10% labeled *MS-COCO 2017 train* (split 1), evaluated on the *MS-COCO 2017 val* set.

3.2. θ: the Confidence Threshold for Hard Labels

In this section we study how the confidence threshold of hard labels affects the final model performance. A reasonable confidence threshold for filtering low-quality pseudo-labels is crucial for hard label models. Here we leave other parameters unchanged and only modify confidence threshold.

The results in Fig. 2 shows that the best threshold is between 0.7 and 0.8. We select $\theta = 0.7$ in the main paper as it leads to the best overall performance among five splits. The result suggests that the soft-label model still outperforms the best hard-label model.

4. Augmentation Details

The strong augmentation we applied to our model follows [5]. The augmentation consists of two operations: one

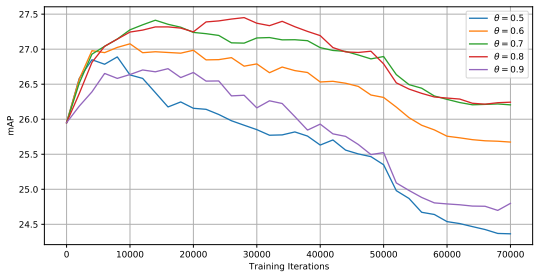

Figure 2: Comparison between hard label models with different θ trained on 10% labeled *MS-COCO 2017 train* (split 1), evaluated on the *MS-COCO 2017 val* set.

operation changes the color, and another operation applies Cutout [1]. The configuration of the first operation is one randomly picked from the following operations, assuming all random numbers are sampled from uniform distributions:

1. Identity: no changes at all.
2. Apply Gaussian blurring with a standard deviation randomly taken from (0, 3).
3. Apply average blurring by computing means over neighbourhoods. The kernel size is randomly picked from (2, 7).
4. Sharpen a image and then alpha-blend the result with the original input image. The blending factor is randomly taken from (0, 1), where 0 means only the original image and 1 means only the sharpen image. The lightness/brightness of the sharpened image is taken from (0.75, 1.5).
5. Apply noise sampled from Gaussian distributions elementwise to the input images. The means of the Gaussian distributions are set to 0. The standard deviations of the Gaussian distributions are sampled from (0, 0.05), which is relative to the maximum pixel value in the image format. Noise is applied on 50% of images per-channel.
6. Invert the color with 5% of probability.
7. Add a value randomly taken from (-10, 10) to 50% of image pixels per channel.
8. Multiply each pixel with a value sampled from (0.5, 1.5). This operation applies to 50% of pixels per channel.
9. Multiply the contrast by a value randomly taken from (0.5, 2) per channel for the given input image.

The second operation is a Cutout. For each image, fill a random number α of cutout square patches on the original image with size either 0 or 0.2 of the input image height. A Cutout patch with size 0 means that particular patch is canceled. α is randomly taken from (1, 5).

References

[1] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 2

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 1, 2

[3] 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*, pages 2117–2125, 2017. 1, 2

[4] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015. 1, 2

[5] 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. 1, 2