

# Adaptive Weighted Co-Learning for Cross-Domain Few-Shot Learning — Supplementary Material

Abdullah Alchihabi<sup>1</sup>  
abdullahalchihabi@gmail.com

Marzi Heidari<sup>1</sup>  
marziheidari@gmail.com

Yuhong Guo<sup>1,2</sup>  
yuhong.guo@carleton.ca

<sup>1</sup> School of Computer Science  
Carleton University  
Ottawa, Canada

<sup>2</sup> Canada CIFAR AI Chair  
Amii, Canada

## 1 Algorithm

The details of the proposed co-learning process are presented in Algorithm 1.

---

### Algorithm 1 The AWCoL Procedure

---

**Input:** Target N-way-K-shot test task  $(S, Q)$ ; source pre-trained models  $M_1$  &  $M_2$

**Output:** Fine-tuned models  $M_1$  &  $M_2$

**Initialization**

**for**  $i = 1$  **to** maxiters **do**

    Choose model  $M_m$  ( $M_1$  or  $M_2$ ) to update

    Calculate class prototypes using support instances

**for**  $x \in Q$  **do**

        Generate WMA prediction vector  $\tilde{\mathbf{p}}^m(x)$

        Calculate the co-learning prediction vector  $\tilde{\mathbf{p}}^{co}(x)$

        Calculate pseudo-label indicator vector  $\hat{\mathbf{y}}_{[x]}$

        Calculate adaptive weight  $w_{[x]}$

        Generate negative pseudo-label vector  $\hat{\mathbf{y}}_{[x]}^N$

**end for**

    Calculate weighted adaptive co-learning loss  $\mathcal{L}_m^{co}$

    Calculate negative pseudo-label loss  $\mathcal{L}_m^N$

    Calculate the total loss  $\mathcal{L}_m$

    Update model  $M_m$  using gradient descent

    Update  $\alpha_m$  for model  $M_m$

**end for**

---

Table 1: Results for using different methods to compute the adaptive weights ( $w_{[x]}$ ) for query instances on 5-way 5-shot CDFSL.

	ChestX	CropDisea.	ISIC	EuroSAT
Average	<b>24.50</b> <sub>(0.33)</sub>	<b>99.59</b> <sub>(0.10)</sub>	<b>58.75</b> <sub>(0.60)</sub>	<b>96.76</b> <sub>(0.27)</sub>
Raw Avg.	23.05 <sub>(0.31)</sub>	98.46 <sub>(0.10)</sub>	56.92 <sub>(0.63)</sub>	94.76 <sub>(0.25)</sub>
Product	23.52 <sub>(0.32)</sub>	99.04 <sub>(0.13)</sub>	54.74 <sub>(0.61)</sub>	92.38 <sub>(0.39)</sub>
Self	22.57 <sub>(0.31)</sub>	97.52 <sub>(0.08)</sub>	50.67 <sub>(0.60)</sub>	91.62 <sub>(0.25)</sub>
Cross	23.05 <sub>(0.31)</sub>	97.49 <sub>(0.09)</sub>	49.93 <sub>(0.62)</sub>	91.74 <sub>(0.26)</sub>

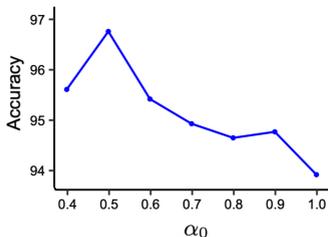
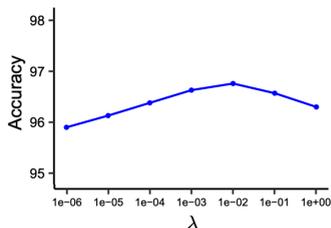
(a)  $\alpha_0$ (b)  $\lambda$ 

Figure 1: Sensitivity analysis for two hyper-parameters  $\alpha_0$  and  $\lambda$  on EuroSAT with cross-domain 5-way 5-shot tasks: (a)  $\alpha_0$ , (b)  $\lambda$ .

## 2 How to Compute the Adaptive Weights?

For the proposed method, we used adaptive weighting to alleviate the negative impact of pseudo-label noise by assigning different weights to different query instances. Here we investigate the effect of our choice of weight calculation by examining five different adaptive weight computation methods: (1) ‘‘Average’’: the proposed weight computation via Eq. (6) and Eq. (8). (2) ‘‘Raw Avg.’’: a variant of the proposed ‘‘Average’’ method that drops the softmax function from Eq. (6) such that  $w_{[x]} = \max(\text{avg}(\tilde{\mathbf{p}}^1(x), \tilde{\mathbf{p}}^2(x)))$ . (3) ‘‘Product’’: it computes the weights from the product of the WMA predictions of both models such that  $w_{[x]} = \max(\text{softmax}(\tilde{\mathbf{p}}^1(x) \times \tilde{\mathbf{p}}^2(x)))$ . (4) ‘‘Self’’: it computes the weights for each model separately from the WMA predictions of itself such that  $w_{[x]}^m = \max(\tilde{\mathbf{p}}^m(x))$ . (5) ‘‘Cross’’: it computes the weights for each model from the WMA predictions of the other model such that  $w_{[x]}^1 = \max(\tilde{\mathbf{p}}^2(x))$  and  $w_{[x]}^2 = \max(\tilde{\mathbf{p}}^1(x))$ . We evaluate the performance of all these five variants of adaptive weighting mechanisms using the 5-way 5-shot cross-domain few-shot learning tasks on four datasets. The results are reported in Table 1.

The table shows that the proposed adaptive weighting mechanism, ‘‘Average’’, clearly outperforms all the other alternatives across all the four datasets. The performance drop of ‘‘Raw Avg.’’ highlights the importance of the softmax renormalization. Meanwhile, the methods that compute the weights from the combined WMA predictions of the two models (‘‘Average’’, ‘‘Raw Avg.’’, and ‘‘Product’’) consistently outperform the methods that use the two models separately (‘‘Self’’ and ‘‘Cross’’). This study validates the choice of our simple average weight computation method.

### 3 Hyper-parameter Sensitivity Analysis

We conduct sensitivity analysis for the proposed AWCoL over two hyper-parameters: the  $\lambda$  trade-off hyper-parameter over the two loss terms and the  $\alpha^0$ —the initial value of the  $\alpha_m$  hyper-parameter for producing WMA predictions. We conduct the experiments on EuroSAT dataset over the cross-domain 5-way 5-shot tasks by testing different values of each of the two hyper-parameters independently. The obtained results are presented in Figure 1.

From the figure we can see that either a too small ( $< 0.001$ ) or too big ( $> 0.1$ )  $\lambda$  value will lead to performance drop and the best result is produced with  $\lambda = 0.01$ . This suggests the negative pseudo-label based loss is helpful as an auxiliary regularizer. As for the hyper-parameter  $\alpha_0$ , larger values ( $> 0.5$ ) cause performance drops. The reason is that larger  $\alpha_0$  allows significantly larger initial updates to the WMA prediction vectors, which might cause oscillating updates. The best result is obtained when  $\alpha_0 = 0.5$ .