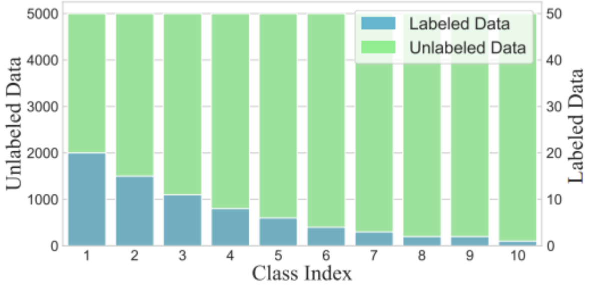
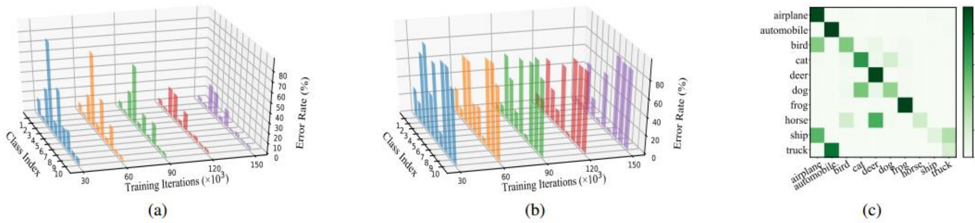
Towards Semi-supervised Learning with Non-random Missing Labels [ICLR 2023]

Challenge: label Missing Not At Random (MNAR)



Training under MNAR, the model increasingly favors some classes, seriously affecting the pseudo-rectifying procedure.

Pseudo-rectifying : the change of the label assignment decision made by the SSL model for the same sample according to the knowledge learned at each new epoch



(a) (b) (c)

Figure 2: Results of FixMatch [27] in MNAR and the conventional SSL setting (*i.e.*, balanced labeled and unlabeled data). The models are trained on CIFAR-10 with WRN-28-2 backbone [37]. (a) and (b): Class-wise pseudo-label error rate. (c): Confusion matrix of pseudo-labels. In (b) and (c), experiments are conducted with the setting of Fig. 1, whereas in (a) with the conventional setting. The label amount used in (a) is the same as that in (b) and (c).

Motivation: The mispredicted pseudo-labels for each class are often concentrated in a few classes, rather than scattered across all other classes. Inspired by this, we argue that it is feasible to guide pseudo-rectifying from the **class level**, i.e., pointing out the latent direction of class transition based on its current class prediction only.

Overview of Pseudo-Rectifying Guidance (PRG):

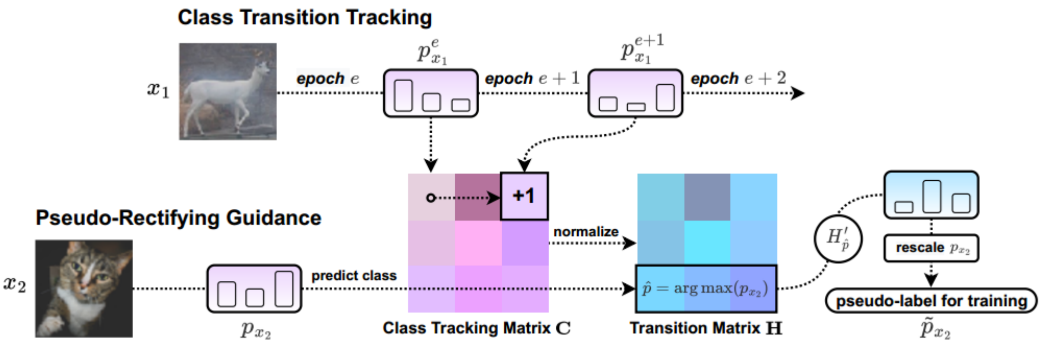

Class Transition Tracking
x_1
epoch e
$p^{e}_{x_1}$
epoch $e+1$
$p^{e+1}_{x_1}$
epoch $e+2$
+1
Pseudo-Rectifying Guidance
normalize
$H'_{\hat{p}}$
rescale p_{x_2}
x_2
predict class
$\hat{p} = \arg\max(p_{x_2})$
pseudo-label for training
p_{x_2}
Class Tracking Matrix $\mathbf{C}$
Transition Matrix $\mathbf{H}$
$\tilde{p}_{x_2}$

Figure 3: Overview of PRG. Class tracking matrix $\mathbf{C}$ is obtained by tracking the class transitions of pseudo-labels (*e.g.*, p_{x_1} for sample x_1) between epoch e and epoch $e+1$ caused by pseudo-rectifying procedure (Eq. (5)). The Markov random walk defined by transition matrix $\mathbf{H}$ (each row H_i represents the transition probability vector corresponding to class i) is modeled on the graph constructed over $\mathbf{C}$. Generally, given a pseudo-label, *e.g.*, p_{x_2} for sample x_2, class- and batch-rescaled $\mathbf{H}$ (*i.e.*, $\mathbf{H}'$) is utilized to provide the class-level pseudo-rectifying guidance for p_{x_2} according to its class prediction $\hat{p} = \arg\max(p_{x_2})$ (Eqs. (6)$\sim$(7)). Finally, the rescaled pseudo-label $\tilde{p}_{x_2}$ is used for the training.

Method:

X: input space
Y: label space (k classes)
M: label missing indicator (m = 1 missing)
p = f(x,θ): pseudo-label

Pseudo-Rectifying Guidance:

Pseudo-rectifying process: the change on p by the next epoch $p^{e+1} = g_\theta(p^e)$

In general, the Pseudo-Rectifying Guidance (PRG) can be described as:

$$\tilde{p}^{e+1} = \text{Normalize}(\eta \circ g_\theta(p^e)), \tag{3}$$

where ∘ is Hadamard product, scaling weight vector η ∈ R^k_+

It is also feasible to guide pseudo-rectifying at class level. Hence, we define rectifying weight matrix $\mathbf{A} \in \mathbb{R}_+^{k\times k}$, where each row A_i is representing the rectifying weight vector corresponding to class i. The class-level pseudo-rectifying guidance:

$$\tilde{p}^{e+1} = \text{Normalize}(A_{\hat{p}^{e+1}} \circ g_\theta(p^e)). \tag{4}$$

Next, we will introduce a simple and feasible way to obtain an effective A for PRG to improve the pseudo-labels.

Class Transition Tracking:

Firstly, we consider building a fully connected graph G in class space Y. This graph is constructed by adjacency matrix C, where each element C_{ij} represents the frequency of class transitions that occur from class i to class j.

$$C_{ij}^{(n)} = \left|\left\{\hat{p}^{(b)} \mid \hat{p}^{(b),e} = i, \hat{p}^{(b),e+1} = j, i \neq j, b \in \{1, ..., B_U\}\right\}\right|, \tag{5}$$

$$n \in \{1, ..., N_B\} \text{ and } C_{ii}^{(n)} = 0$$

$$C_{ij} = \textstyle\sum_{n=1}^{N_B} C_{ij}^{(n)} / N_B$$

Hereafter, we define the Markov random walk along the nodes of G, which is characterized by its transition matrix H. and H_{ij} represents the transition probability for the class prediction transits from class i at epoch e to class j at epoch e + 1.

Since H_{ii} = 0 is unreasonable, we control the probability that does not transition class by setting $H_{ii} = \frac{\alpha}{k-1}$.

In addition, to better provide class-level guidance, we scale each element in H by

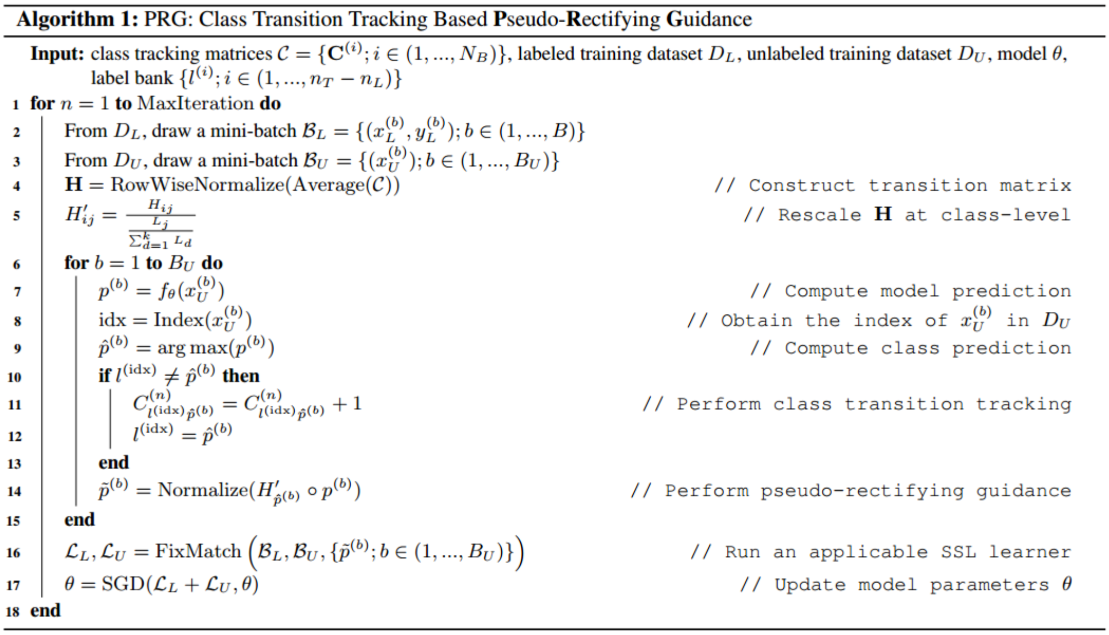
$$H'_{ij} = \frac{H_{ij}}{\frac{L_j}{\sum_{d=1}^{k} L_d}} \tag{6}$$

where L ∈ R^{k}_{+} and L_i records the number of class predictions belonging to class i averaged on last N_B batches.

Finally, we have

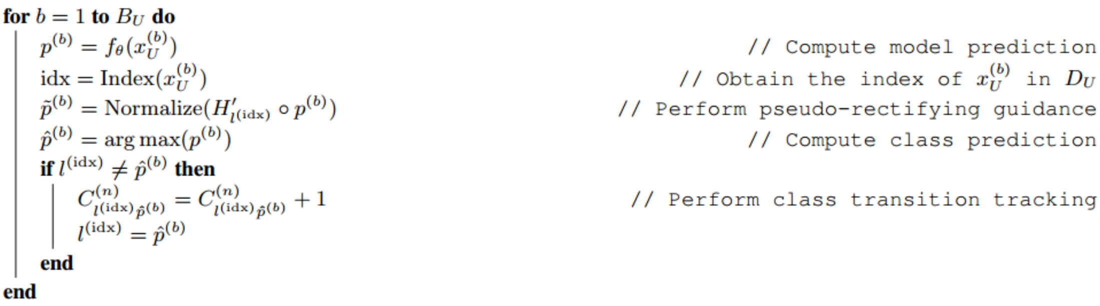
$$\tilde{p}^{e+1} = \mathrm{Normalize}(H'_{\hat{p}^{e+1}} \circ g_\theta(p^e)), \tag{7}$$

It is also feasible to use the class transition driven by $\widehat{p^e}$ to revise p^{e+1}.



Algorithm 1: PRG: Class Transition Tracking Based **P**seudo-**R**ectifying **G**uidance

```
Input: class tracking matrices $\mathcal{C} = \{\mathbf{C}^{(i)}; i \in (1, ..., N_B)\}$, labeled training dataset $D_L$, unlabeled training dataset $D_U$, model $\theta$,
       label bank $\{l^{(i)}; i \in (1, ..., n_T - n_L)\}$
for $n = 1$ to MaxIteration do
    From $D_L$, draw a mini-batch $\mathcal{B}_L = \{(x_L^{(b)}, y_L^{(b)}); b \in (1, ..., B)\}$
    From $D_U$, draw a mini-batch $\mathcal{B}_U = \{(x_U^{(b)}); b \in (1, ..., B_U)\}$
    $\mathbf{H} = \mathrm{RowWiseNormalize}(\mathrm{Average}(\mathcal{C}))$            // Construct transition matrix
    $H'_{ij} = \frac{H_{ij}}{\frac{L_j}{\sum_{d=1}^{k} L_d}}$                              // Rescale $\mathbf{H}$ at class-level
    for $b = 1$ to $B_U$ do
        $p^{(b)} = f_\theta(x_U^{(b)})$                                 // Compute model prediction
        $\mathrm{idx} = \mathrm{Index}(x_U^{(b)})$                            // Obtain the index of $x_U^{(b)}$ in $D_U$
        $\hat{p}^{(b)} = \arg\max(p^{(b)})$                             // Compute class prediction
        if $l^{(\mathrm{idx})} \neq \hat{p}^{(b)}$ then
            $C^{(n)}_{l^{(\mathrm{idx})}\hat{p}^{(b)}} = C^{(n)}_{l^{(\mathrm{idx})}\hat{p}^{(b)}} + 1$            // Perform class transition tracking
            $l^{(\mathrm{idx})} = \hat{p}^{(b)}$
        end
        $\tilde{p}^{(b)} = \mathrm{Normalize}(H'_{\hat{p}^{(b)}} \circ p^{(b)})$               // Perform pseudo-rectifying guidance
    end
    $\mathcal{L}_L, \mathcal{L}_U = \mathrm{FixMatch}\left(\mathcal{B}_L, \mathcal{B}_U, \{\tilde{p}^{(b)}; b \in (1, ..., B_U)\}\right)$     // Run an applicable SSL learner
    $\theta = \mathrm{SGD}(\mathcal{L}_L + \mathcal{L}_U, \theta)$                              // Update model parameters $\theta$
end
```



```
for $b = 1$ to $B_U$ do
    $p^{(b)} = f_\theta(x_U^{(b)})$                                     // Compute model prediction
    $\mathrm{idx} = \mathrm{Index}(x_U^{(b)})$                                // Obtain the index of $x_U^{(b)}$ in $D_U$
    $\tilde{p}^{(b)} = \mathrm{Normalize}(H'_{l^{(\mathrm{idx})}} \circ p^{(b)})$           // Perform pseudo-rectifying guidance
    $\hat{p}^{(b)} = \arg\max(p^{(b)})$                                 // Compute class prediction
    if $l^{(\mathrm{idx})} \neq \hat{p}^{(b)}$ then
        $C^{(n)}_{l^{(\mathrm{idx})}\hat{p}^{(b)}} = C^{(n)}_{l^{(\mathrm{idx})}\hat{p}^{(b)}} + 1$            // Perform class transition tracking
        $l^{(\mathrm{idx})} = \hat{p}^{(b)}$
    end
end
```