REMIXMATCH: Semi-supervised learning with distribution alignment and augmentation anchoring [ICLR 2020]

Distribution Alignment: the marginal distribution of predictions on D_u to be close to ~~~~~~~~ ground truth labels

Augmentation Anchoring: Feed mulitiple Strongly augmented versions ("CTAugment") of an input encourage each output to be closed to the weakly augmentation.

MixMatch:

$$\mathcal{X} = \{(x_b, p_b) : b \in (1, \cdots, B)\}$$

$$\mathcal{U} = \{u_b : b \in (1, \cdots, B)\} \xrightarrow{u_b} \hat{u}_{b,k}, k \in \{1, \cdots, K\} \xrightarrow{\text{average}} \bar{q}_b = \frac{1}{K}\sum_k p(y \mid \hat{u}_{b,k}, \theta)$$

K weakly augmentation

$\downarrow$ Sharpen

guess labels

$$\mathcal{U}_g = \{(u_b, q_b)\}$$

Combine $\mathcal{X}$ and $\mathcal{U}_g \rightarrow \mathcal{Y}$

MixUp: $(x', p') = \lambda(x_1, p_1) + (1-\lambda)(x_2, p_2), \forall (x_1, p_1), (x_2, p_2) \in \mathcal{Y}$

Given these mixed-up samples, it performs standrad fully-supervised training with minor modifications.

ReMixMatch:

① Distribution Alignment

Input-Output mutual information: (maximize)

$$I(y;x) = \iint p(y,x) \log \frac{p(y,x)}{p(y)\,p(x)}\, dy\, dx$$

$$= \underbrace{\mathcal{H}(\mathbb{E}_x[p_{model}(y|x;\theta)])}_{} - \underbrace{\mathbb{E}_x[\mathcal{H}(p_{model}(y|x;\theta))]}_{}$$

dataset's marginal class distribution p(y) uniform ✗ "fair"

entropy minimization (high confidence in a class label)

maintain a running average of the model's prediction on $\mathcal{U}$ $\tilde{p}(y)$

given the model's prediction $q = p_{model}(y|u;\theta)$

scale q by a ratio and renormalize the result

$$\tilde{q} = \text{Normalize}\left(q \times \frac{p(y)}{\tilde{p}(y)}\right)$$

eg. $q = (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$, $\tilde{p}(y) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, $p(y) = (1, 0, 0)$

$$\tilde{q} = \text{Normalize}\left((\tfrac{1}{2}, \tfrac{1}{4}, \tfrac{1}{4}) \times (\tfrac{1}{3}, 0, 0)\right) = (1, 0, 0)$$

Note that $p(y)$ is estimated by the labeled examples seen during training. Or it is known a priori.

② Augmentation Anchoring

An unlabeled input $\xrightarrow{\text{weakly}}$ "Anchor"

An unlabeled input $\xrightarrow[\text{(CTAugment)}]{\text{K strongly}}$ K Augmented images $\xrightarrow[\text{Sharpen}]{\text{DA}}$ K targets

Cross-entropy

Control Theory Augment (CTAugment)

A method for learning a data augmentation policy which results in high validation set accuracy.

Algorithm 1 ReMixMatch algorithm for producing a collection of processed labeled examples and processed unlabeled examples with label guesses (cf. Berthelot et al. (2019) Algorithm 1.)

Input: Batch of labeled examples and their one-hot labels $\mathcal{X} = \{(x_b, p_b) : b \in (1, \ldots, B)\}$, batch of unlabeled examples $\mathcal{U} = \{u_b : b \in (1, \ldots, B)\}$, sharpening temperature T, number of augmentations K, Beta distribution parameter α for MixUp.
for $b = 1$ **to** B **do**
  $\hat{x}_b = \text{StrongAugment}(x_b)$ *// Apply strong data augmentation to x_b*
  $\hat{u}_{b,k} = \text{StrongAugment}(u_b); k \in \{1, \ldots, K\}$ *// Apply strong data augmentation K times to u_b*
  $\tilde{u}_b = \text{WeakAugment}(u_b)$ *// Apply weak data augmentation to u_b*
  $q_b = p_{\text{model}}(y \mid \tilde{u}_b; \theta)$ *// Compute prediction for weak augmentation of u_b*
  $q_b = \text{Normalize}(q_b \times p(y)/\tilde{p}(y))$ *// Apply distribution alignment*
  $q_b = \text{Normalize}(q_b^{1/T})$ *// Apply temperature sharpening to label guess*
end for
$\hat{\mathcal{X}} = \big((\hat{x}_b, p_b); b \in (1, \ldots, B)\big)$ *// Augmented labeled examples and their labels*
$\hat{\mathcal{U}}_1 = \big((\hat{u}_{b,1}, q_b); b \in (1, \ldots, B)\big)$ *// First strongly augmented unlabeled example and guessed label*
$\hat{\mathcal{U}} = \big((\hat{u}_{b,k}, q_b); b \in (1, \ldots, B), k \in (1, \ldots, K)\big)$ *// All strongly augmented unlabeled examples*
$\hat{\mathcal{U}} = \hat{\mathcal{U}} \cup \big((\tilde{u}_b, q_b); b \in (1, \ldots, B)\big)$ *// Add weakly augmented unlabeled examples*
$\mathcal{W} = \text{Shuffle}\big(\text{Concat}(\hat{\mathcal{X}}, \hat{\mathcal{U}})\big)$ *// Combine and shuffle labeled and unlabeled data*
$\mathcal{X}' = \big(\text{MixUp}(\hat{\mathcal{X}}_i, \mathcal{W}_i); i \in (1, \ldots, |\hat{\mathcal{X}}|)\big)$ *// Apply MixUp to labeled data and entries from $\mathcal{W}$*
$\mathcal{U}' = \big(\text{MixUp}(\hat{\mathcal{U}}_i, \mathcal{W}_{i+|\hat{\mathcal{X}}|}); i \in (1, \ldots, |\hat{\mathcal{U}}|)\big)$ *// Apply MixUp to unlabeled data and the rest of $\mathcal{W}$*
return $\mathcal{X}', \mathcal{U}', \underline{\hat{\mathcal{U}}_1}$

↳ two additional loss

Pre-mixup unlabeled loss We feed the guessed labels and predictions for example in $\hat{\mathcal{U}}_1$ as-is into a separate cross-entropy loss term.

Rotation loss Recent result have shown that applying ideas from self-supervised learning to SSL can produce strong performance (Gidaris et al., 2018; Zhai et al., 2019). We integrate this idea by rotating each image $u \in \hat{\mathcal{U}}_1$ as $\mathrm{Rotate}(u, r)$ where we sample the rotation angle r uniformly from $r \sim \{0, 90, 180, 270\}$ and then ask the model to predict the rotation amount as a four-class classification problem.

In total, the ReMixMatch loss is

$$\sum_{x,p \in \mathcal{X}'} \mathrm{H}(p, p_{\text{model}}(y|x;\theta)) + \lambda_{\mathcal{U}} \sum_{u,q \in \mathcal{U}'} \mathrm{H}(q, p_{\text{model}}(y|u;\theta)) \tag{3}$$

$$+\lambda_{\hat{\mathcal{U}}_1} \sum_{u,q \in \hat{\mathcal{U}}_1} \mathrm{H}(q, p_{\text{model}}(y|u;\theta)) + \lambda_r \sum_{u \in \hat{\mathcal{U}}_1} \mathrm{H}(r, p_{\text{model}}(r|\,\mathrm{Rotate}(u,r);\theta)) \tag{4}$$