DIVIDEMIX: LEARNING WITH NOISY LABELS AS SEMI-SUPERVISED LEARNING

Junnan Li, Richard Socher, Steven C.H. Hoi

Salesforce Research

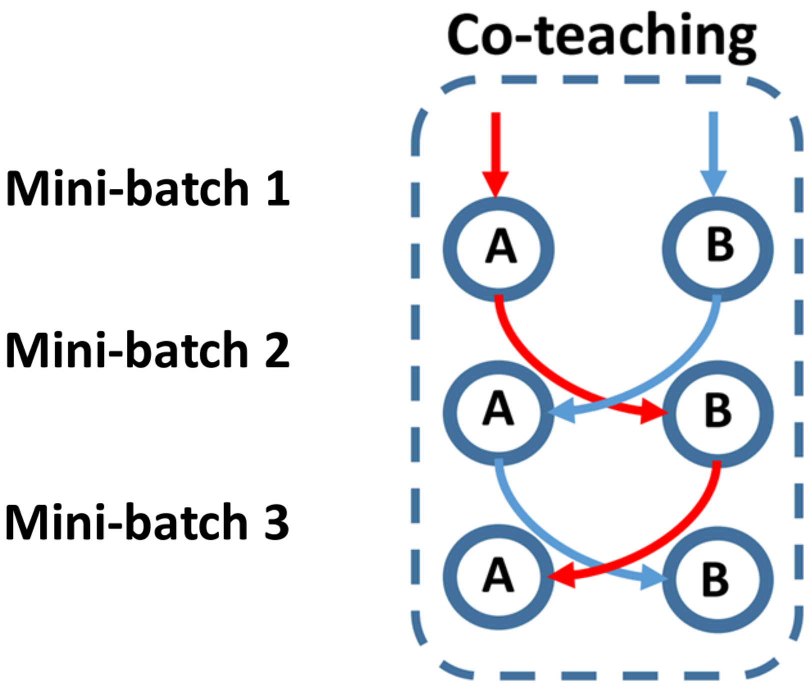
ICLR 2020

Contents

- Co-teaching
- MixMatch
- Methods
- Experiments

Co-teaching

- In each mini-batch of data, each network views its **small-loss instances** as the useful knowledge, and teaches such useful instances to its peer network for updating the parameters.



MixMatch

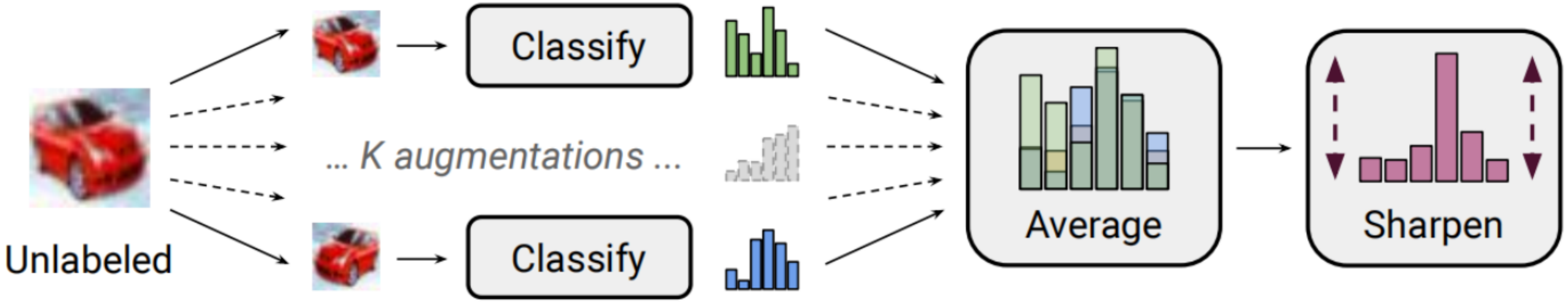
■ **Data Augmentation**

$$\hat{x}_b = \text{Augment}(x_b)$$

$$\hat{u}_{b,k} = \text{Augment}(u_b), k \in (1, \ldots, K)$$ K augmentations

■ **Label Guessing**

$$\bar{q}_b = \frac{1}{K}\sum_{k=1}^{K} \mathrm{p}_{\text{model}}(y \mid \hat{u}_{b,k}; \theta) \qquad \text{Sharpen}(p, T)_i := p_i^{\frac{1}{T}} \Big/ \sum_{j=1}^{L} p_j^{\frac{1}{T}}$$



MixMatch

■ MixUp

Mix both labeled examples and unlabeled examples with label guesses
Corresponding labels probabilities $(x1, p1), (x2, p2)$ we compute $(x`, p`)$

$$\lambda \sim \mathrm{Beta}(\alpha, \alpha)$$

$$\lambda' = \max(\lambda, 1 - \lambda)$$

$$x' = \lambda' x_1 + (1 - \lambda') x_2$$

$$p' = \lambda' p_1 + (1 - \lambda') p_2$$

$\hat{\mathcal{X}} = \big((\hat{x}_b, p_b); b \in (1, \ldots, B)\big)$ *// Augmented labeled examples and their labels*
$\hat{\mathcal{U}} = \big((\hat{u}_{b,k}, q_b); b \in (1, \ldots, B), k \in (1, \ldots, K)\big)$ *// Augmented unlabeled examples, guessed labels*
$\mathcal{W} = \mathrm{Shuffle}\big(\mathrm{Concat}(\hat{\mathcal{X}}, \hat{\mathcal{U}})\big)$ *// Combine and shuffle labeled and unlabeled data*
$\mathcal{X}' = \big(\mathrm{MixUp}(\hat{\mathcal{X}}_i, \mathcal{W}_i); i \in (1, \ldots, |\hat{\mathcal{X}}|)\big)$ *// Apply* MixUp *to labeled data and entries from* $\mathcal{W}$
$\mathcal{U}' = \big(\mathrm{MixUp}(\hat{\mathcal{U}}_i, \mathcal{W}_{i+|\hat{\mathcal{X}}|}); i \in (1, \ldots, |\hat{\mathcal{U}}|)\big)$ *// Apply* MixUp *to unlabeled data and the rest of* $\mathcal{W}$

MixMatch

- **Loss Function**

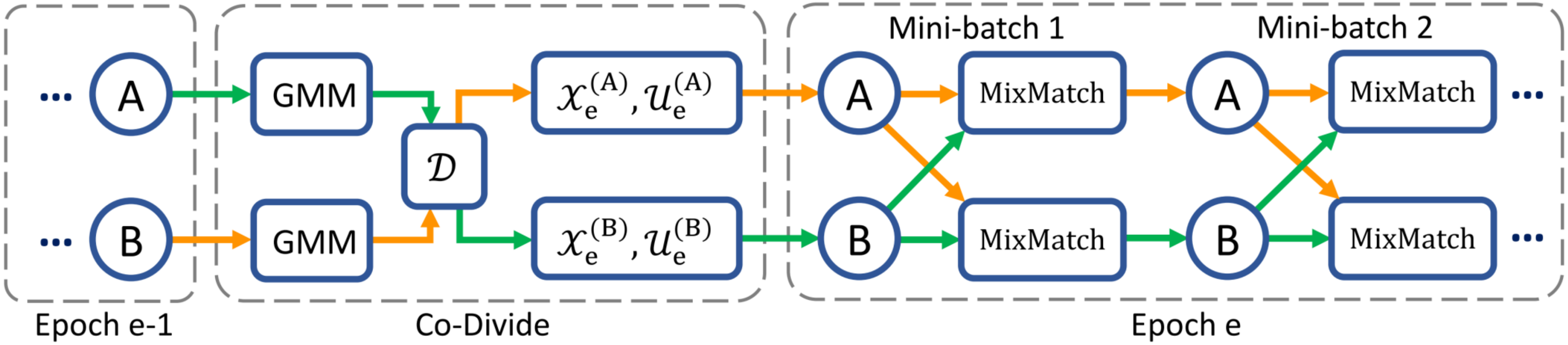
$$\mathcal{X}', \mathcal{U}' = \mathrm{MixMatch}(\mathcal{X}, \mathcal{U}, T, K, \alpha)$$

$$\mathcal{L}_{\mathcal{X}} = \frac{1}{|\mathcal{X}'|} \sum_{x,p \in \mathcal{X}'} \mathrm{H}(p, \mathrm{p}_{\mathrm{model}}(y \mid x; \theta))$$

$$\mathcal{L}_{\mathcal{U}} = \frac{1}{L|\mathcal{U}'|} \sum_{u,q \in \mathcal{U}'} \|q - \mathrm{p}_{\mathrm{model}}(y \mid u; \theta)\|_2^2$$

$$\mathcal{L} = \mathcal{L}_{\mathcal{X}} + \lambda_{\mathcal{U}} \mathcal{L}_{\mathcal{U}}$$

Methods



Mini-batch 1
Mini-batch 2
A
B
GMM
GMM
$\mathcal{D}$
$\mathcal{X}_e^{(A)}, \mathcal{U}_e^{(A)}$
$\mathcal{X}_e^{(B)}, \mathcal{U}_e^{(B)}$
A
B
MixMatch
MixMatch
A
B
MixMatch
MixMatch
Epoch e-1
Co-Divide
Epoch e

Co-Divide

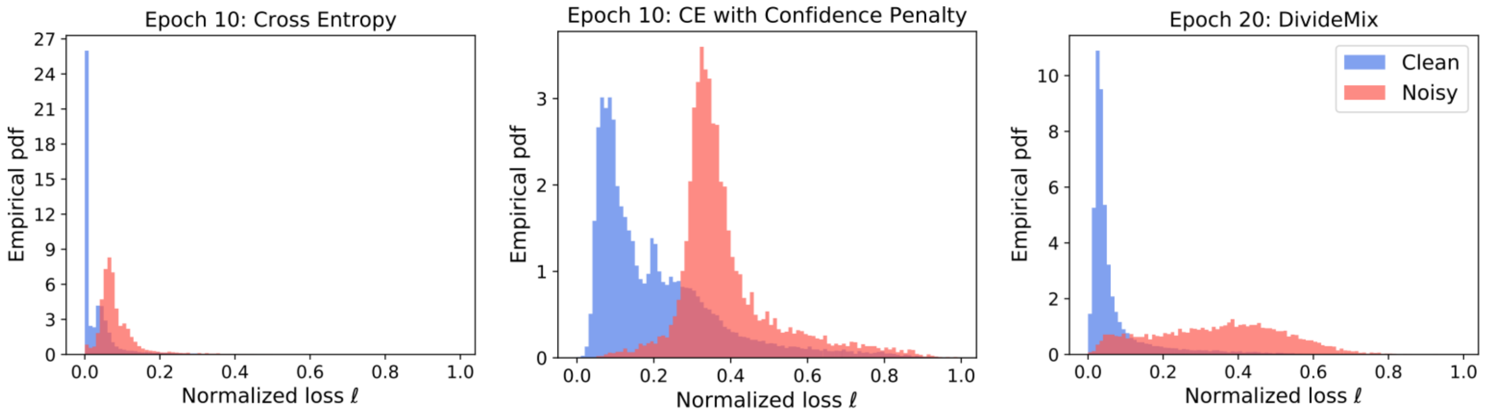
- “Warm up” the model for a few epochs by training on all data using the standard cross-entropy loss
- For each network, fit a two-component GMM to ℓ using the EM algorithm.
- For each sample, its clean probability w_i is the posterior probability $p(g \mid \ell_i)$
- Divide the training data into a labeled set and an unlabeled set by setting a threshold τ on w_i
- Feed the data to each other

Co-Divide

-Confidence Penalty for Asymmetric Noise

- The GMM cannot effectively distinguish clean and noisy samples based on the loss distribution
- To address this issue, we penalize confident predictions from the network by adding a **negative entropy term** (Pereyra et al., 2017)

$$\mathcal{H} = -\sum_{c} \mathrm{p}^{\mathrm{c}}_{\mathrm{model}}(x;\theta)\log\left(\mathrm{p}^{\mathrm{c}}_{\mathrm{model}}(x;\theta)\right)$$



(a) (b) (c)

MIXMATCH

- Train the two networks one at a time while keeping the other one fixed.
- Perform label co-refinement for labeled samples

$$\overline{y}_b = w_b y_b + (1 - w_b) p_b$$

- Sharpening function on the refined label

$$\hat{y}_b = \mathrm{Sharpen}(\bar{y}_b, T) = {\bar{y}_b^c}^{\frac{1}{T}} \Big/ \sum_{c=1}^{C} {\bar{y}_b^c}^{\frac{1}{T}}, \text{ for } c = 1, 2, ..., C.$$

- Use the ensemble of predictions from both networks to “co-guess” the labels for unlabeled samples

MIXMATCH

- Loss

$$\mathcal{L}_{\mathcal{X}} = -\frac{1}{|\mathcal{X}'|} \sum_{x,p \in \mathcal{X}'} \sum_{c} p_c \log\left(\mathrm{p}^{\mathrm{c}}_{\mathrm{model}}\,(x;\theta)\right) \quad \mathcal{L}_{\mathcal{U}} = \frac{1}{|\mathcal{U}'|} \sum_{x,p \in \mathcal{U}'} \|p - \mathrm{p}_{\mathrm{model}}(x;\theta)\|_2^2$$

- To prevent assigning all samples to a single class, we apply the regularization term

$$\mathcal{L}_{\mathrm{reg}} = \sum_{c} \pi_c \log\left(\pi_c \Big/ \frac{1}{|\mathcal{X}'| + |\mathcal{U}'|} \sum_{x \in \mathcal{X}' + \mathcal{U}'} \mathrm{p}^{\mathrm{c}}_{\mathrm{model}}\,(x;\theta)\right)$$

- The total loss is

$$\mathcal{L} = \mathcal{L}_{\mathcal{X}} + \lambda_u \mathcal{L}_{\mathcal{U}} + \lambda_r \mathcal{L}_{\mathrm{reg}}$$

Algorithm 1: DivideMix. Line 4-8: co-divide; Line 17-18: label co-refinement; Line 20: label co-guessing.

1 **Input:** $\theta^{(1)}$ and $\theta^{(2)}$, training dataset $(\mathcal{X}, \mathcal{Y})$, clean probability threshold τ, number of augmentations M, sharpening temperature T, unsupervised loss weight λ_u, Beta distribution parameter α for MixMatch.
2 $\theta^{(1)}, \theta^{(2)} = \text{WarmUp}(\mathcal{X}, \mathcal{Y}, \theta^{(1)}, \theta^{(2)})$ *// standard training (with confidence penalty)*
3 **while** $e < \text{MaxEpoch}$ **do**
4   $\mathcal{W}^{(2)} = \text{GMM}(\mathcal{X}, \mathcal{Y}, \theta^{(1)})$ *// model per-sample loss with $\theta^{(1)}$ to obtain clean proabability for $\theta^{(2)}$*
5   $\mathcal{W}^{(1)} = \text{GMM}(\mathcal{X}, \mathcal{Y}, \theta^{(2)})$ *// model per-sample loss with $\theta^{(2)}$ to obtain clean proabability for $\theta^{(1)}$*
6   **for** $k = 1, 2$ **do** *// train the two networks one by one*
7     $\mathcal{X}_e^{(k)} = \{(x_i, y_i, w_i) | w_i \geq \tau, \forall (x_i, y_i, w_i) \in (\mathcal{X}, \mathcal{Y}, \mathcal{W}^{(k)})\}$ *// labeled training set for $\theta^{(k)}$*
8     $\mathcal{U}_e^{(k)} = \{x_i | w_i < \tau, \forall (x_i, w_i) \in (\mathcal{X}, \mathcal{W}^{(k)})\}$ *// unlabeled training set for $\theta^{(k)}$*
9   **for** iter $= 1$ **to** num_iters **do**
10   From $\mathcal{X}_e^{(k)}$, draw a mini-batch $\{(x_b, y_b, w_b); b \in (1, ..., B)\}$
11   From $\mathcal{U}_e^{(k)}$, draw a mini-batch $\{u_b; b \in (1, ..., B)\}$
12   **for** $b = 1$ **to** B **do**
13   **for** $m = 1$ **to** M **do**
14           $\hat{x}_{b,m} = \text{Augment}(x_b)$ *// apply m^{th} round of augmentation to x_b*
15           $\hat{u}_{b,m} = \text{Augment}(u_b)$ *// apply m^{th} round of augmentation to u_b*
16   **end**
17         $p_b = \frac{1}{M} \sum_m \text{p}_{\text{model}}(\hat{x}_{b,m}; \theta^{(k)})$ *// average the predictions across augmentations of x_b*
18         $\bar{y}_b = w_b y_b + (1 - w_b) p_b$
          // refine ground-truth label guided by the clean probability produced by the other network
19         $\hat{y}_b = \text{Sharpen}(\bar{y}_b, T)$ *// apply temperature sharpening to the refined label*
20         $\bar{q}_b = \frac{1}{2M} \sum_m \left(\text{p}_{\text{model}}(\hat{u}_{b,m}; \theta^{(1)}) + \text{p}_{\text{model}}(\hat{u}_{b,m}; \theta^{(2)})\right)$
          // co-guessing: average the predictions from both networks across augmentations of u_b
21         $q_b = \text{Sharpen}(\bar{q}_b, T)$ *// apply temperature sharpening to the guessed label*
22   **end**
23       $\hat{\mathcal{X}} = \{(\hat{x}_{b,m}, \hat{y}_b); b \in (1, ..., B), m \in (1, ..., M)\}$ *// augmented labeled mini-batch*
24       $\hat{\mathcal{U}} = \{(\hat{u}_{b,m}, q_b); b \in (1, ..., B), m \in (1, ..., M)\}$ *// augmented unlabeled mini-batch*
25       $\mathcal{L}_{\mathcal{X}}, \mathcal{L}_{\mathcal{U}} = \text{MixMatch}(\hat{\mathcal{X}}, \hat{\mathcal{U}})$ *// apply MixMatch*
26       $\mathcal{L} = \mathcal{L}_{\mathcal{X}} + \lambda_u \mathcal{L}_{\mathcal{U}} + \lambda_r \mathcal{L}_{\text{reg}}$ *// total loss*
27       $\theta^{(k)} = \text{SGD}(\mathcal{L}, \theta^{(k)})$ *// update model parameters*
28   **end**
29   **end**
30 **end**

CIFAR-10 & CIFAR-100

Dataset		CIFAR-10				CIFAR-100			
Method/Noise ratio		20%	50%	80%	90%	20%	50%	80%	90%
Cross-Entropy	Best	86.8	79.4	62.9	42.7	62.0	46.7	19.9	10.1
	Last	82.7	57.9	26.1	16.8	61.8	37.3	8.8	3.5
Bootstrap (Reed et al., 2015)	Best	86.8	79.8	63.3	42.9	62.1	46.6	19.9	10.2
	Last	82.9	58.4	26.8	17.0	62.0	37.9	8.9	3.8
F-correction (Patrini et al., 2017)	Best	86.8	79.8	63.3	42.9	61.5	46.6	19.9	10.2
	Last	83.1	59.4	26.2	18.8	61.4	37.3	9.0	3.4
Co-teaching+* (Yu et al., 2019)	Best	89.5	85.7	67.4	47.9	65.6	51.8	27.9	13.7
	Last	88.2	84.1	45.5	30.1	64.1	45.3	15.5	8.8
Mixup (Zhang et al., 2018)	Best	95.6	87.1	71.6	52.2	67.8	57.3	30.8	14.6
	Last	92.3	77.6	46.7	43.9	66.0	46.6	17.6	8.1
P-correction* (Yi & Wu, 2019)	Best	92.4	89.1	77.5	58.9	69.4	57.5	31.1	15.3
	Last	92.0	88.7	76.5	58.2	68.1	56.4	20.7	8.8
Meta-Learning* (Li et al., 2019)	Best	92.9	89.3	77.4	58.7	68.5	59.2	42.4	19.5
	Last	92.0	88.8	76.1	58.3	67.7	58.0	40.1	14.3
M-correction (Arazo et al., 2019)	Best	94.0	92.0	86.8	69.1	73.9	66.1	48.2	24.3
	Last	93.8	91.9	86.6	68.7	73.4	65.4	47.6	20.5
DivideMix	Best	**96.1**	**94.6**	**93.2**	**76.0**	**77.3**	**74.6**	**60.2**	**31.5**
	Last	**95.7**	**94.4**	**92.9**	**75.4**	**76.9**	**74.2**	**59.6**	**31.0**

Table 1: Comparison with state-of-the-art methods in test accuracy (%) on CIFAR-10 and CIFAR-100 with symmetric noise. Methods marked by * denote re-implementations based on public code.

Method	Best	Last
Cross-Entropy	85.0	72.3
F-correction (Patrini et al., 2017)	87.2	83.1
M-correction (Arazo et al., 2019)	87.4	86.3
Iterative-CV (Chen et al., 2019)	88.6	88.0
P-correction (Yi & Wu, 2019)	88.5	88.1
Joint-Optim (Tanaka et al., 2018)	88.9	88.4
Meta-Learning (Li et al., 2019)	89.2	88.6
DivideMix	**93.4**	**92.1**

Table 2: Comparison with state-of-the-art methods in test accuracy (%) on CIFAR-10 with 40% **asymmetric noise**.

Clothing1M & WebVision

Method	WebVision		ILSVRC12	
	top1	top5	top1	top5
F-correction (Patrini et al., 2017)	61.12	82.68	57.36	82.36
Decoupling (Malach & Shalev-Shwartz, 2017)	62.54	84.74	58.26	82.26
D2L (Ma et al., 2018)	62.68	84.00	57.80	81.36
MentorNet (Jiang et al., 2018)	63.00	81.40	57.80	79.92
Co-teaching (Han et al., 2018)	63.58	85.20	61.48	84.70
Iterative-CV (Chen et al., 2019)	65.24	85.34	61.60	84.98
DivideMix	**77.32**	**91.64**	**75.20**	**90.84**

Table 3:Comparison with state-of-the-art methods in test accuracy (%) on Clothing1M. Results for baselines are copied from original papers.

Method	Test Accuracy
Cross-Entropy	69.21
F-correction (Patrini et al., 2017)	69.84
M-correction (Arazo et al., 2019)	71.00
Joint-Optim (Tanaka et al., 2018)	72.16
Meta-Cleaner (Zhang et al., 2019)	72.50
Meta-Learning (Li et al., 2019)	73.47
P-correction (Yi & Wu, 2019)	73.49
DivideMix	**74.76**

Table 4: Comparison with state-of-the-art methods trained on (mini) WebVision dataset. Numbers denote top-1 (top-5) accuracy (%) on the WebVision validation set and the ImageNet ILSVRC12 validation set. Results for baseline methods are copied from Chen et al. (2019).

ABLATION STUDY

Dataset		CIFAR-10					CIFAR-100			
Noise type		Sym.				Asym.	Sym.			
Methods/Noise ratio		20%	50%	80%	90%	40%	20%	50%	80%	90%
DivideMix	Best	**96.1**	**94.6**	**93.2**	**76.0**	**93.4**	**77.3**	**74.6**	**60.2**	**31.5**
	Last	**95.7**	**94.4**	**92.9**	**75.4**	**92.1**	**76.9**	**74.2**	**59.6**	**31.0**
DivideMix with $\theta^{(1)}$ test	Best	95.2	94.2	93.0	75.5	92.7	75.2	72.8	58.3	29.9
	Last	95.0	93.7	92.4	74.2	91.4	74.8	72.1	57.6	29.2
DivideMix w/o co-training	Best	95.0	94.0	92.6	74.3	91.9	74.8	72.3	56.7	27.7
	Last	94.8	93.3	92.2	73.2	90.6	74.1	71.7	56.3	27.2
DivideMix w/o label refinement	Best	96.0	94.6	93.0	73.7	87.7	76.9	74.2	58.7	26.9
	Last	95.5	94.2	92.7	73.0	86.3	76.4	73.9	58.2	26.3
DivideMix w/o augmentation	Best	95.3	94.1	92.2	73.9	89.5	76.5	73.1	58.2	26.9
	Last	94.9	93.5	91.8	73.0	88.4	76.2	72.6	58.0	26.4
Divide and MixMatch	Best	94.1	92.8	89.7	70.1	86.5	73.7	70.5	55.3	25.0
	Last	93.5	92.3	89.1	68.6	85.2	72.4	69.7	53.9	23.7

Table 5: Ablation study results in terms of test accuracy (%) on CIFAR-10 and CIFAR-100.