Me-Momentum Supplementary

Yingbin Bai Tongliang Liu*
Trustworthy Machine Learning Lab, University of Sydney

1. Generating label noise

The label noise is generated according to symmetric class-dependent and instance-dependent noise transition matrices.

Noise transition matrix The *noise transition matrix* $T(x)$ was proposed to explicitly model the generation process of label noise, where $T_{ij}(x) = \Pr(\bar{Y} = j|Y = i, X = x)$, $\Pr(A)$ denotes as the probability of the event A, X as the random variable for the instance, $\bar{Y}$ as the noisy label, and Y as the latent clean label. At a high level, the ij-th entry of the transition matrix denotes the probability that the instance will flip from the clean class j to the noisy class i.

Symmetric class-dependent label noise If the flip rate is α, the diagonal entries of a symmetric transition matrix are $1 - \alpha$ and the off-diagonal entries are $\alpha/(c - 1)$ [2].

Instance-dependent label noise We generate the instance-dependent label noise according to the following Algorithm 1 [8].

Algorithm 1 Instance-dependent Label Noise Generation

Input: Clean samples $\{(x_i, y_i)\}_{i=1}^n$; Noise rate τ.
1: Sample instance flip rates $q \in \mathbb{R}^n$ from the truncated normal distribution $\tau\mathcal{N}(\tau, 0.1^2, [0, 1])$;
2: Independently sample $w_1, w_2, \ldots, w_c$ from the standard normal distribution $\mathcal{N}(0, 1^2)$;
3: For $i = 1, 2, \ldots, n$ do
4: $p = x_i \times w_{y_i}$; // generate instance-dependent flip rates
5: $p_{y_i} = -\infty$; // control the diagonal entry of the instance-dependent transition matrix
6: $p = q_i \times softmax(p)$; // make the sum of the off-diagonal entries of the y_i-th row to be q_i
7: $p_{y_i} = 1 - q_i$; // set the diagonal entry to be $1 - q_i$
8: Randomly choose a label from the label space according to possibilities p as noisy label $\bar{y}_i$;
9: End for.
Output: Noisy samples $\{(x_i, \bar{y}_i)\}_{i=1}^n$

*Correspondence to Tongliang Liu (tongliang.liu@sydney.edu.au).

2. Compare results with SELF

The performance of our re-implemented SELF [6] is not as good as that in the original paper. For a fair comparison, we change our backbone consistently with SELF and compare with the results from the original paper directly (without Mean Teachers [7]). Specifically, the network is changed to ResNet26 with Shake-shake regularization [1]. The learning rate is set to 0.05 with weight decay of 2e-4.

Table 1 and Table 2 show that Me-Momentum outperforms SELF by a large margin in *CIFAR10* and *CIFAR100* with Symmetric 40% and Symmetric 60% noise. The gap between the performance of Me-Momentum and SELF becomes larger in Symmetric 60% in both datasets compared with Symmetric 40% because hard confident examples play a more important role in more noisy examples. Specifically, both of the methods are based on the same backbone with Cross-Entropy loss, so the improvement of Me-Momentum can only be as a result of the quality of extracted confident examples. Therefore, Me-Momentum is able to extract better hard confident examples than SELF.

Table 1. Means and standard deviations of classification accuracy compared with SELF on *CIFAR10*

	Sym-40	Sym-60
SELF	87.35%	75.47%
Ours	**92.31%**	**87.88%**

Table 2. Means and standard deviations of classification accuracy compared with SELF on *CIFAR100*

	Sym-40	Sym-60
SELF	61.40%	50.60%
Ours	**68.25%**	**59.51%**

Furthermore, the performance of Me-Momentum can be improved by changing a better backbone. However, to show the effectiveness of the proposed method and avoid complexity, we choose the standard CNN network in the paper.

3. Experiments on high noise rates

In Table 3, we evaluate our method on Symmetric 50%. Note that we only make use of confident examples and discard the non-confident examples. If the noise rate is too

Table 3. Means and standard deviations of classification accuracy on symmetric 50% label noise with different datasets

Flipping-Rate	Cross-Entropy	MentorNet	Co-teaching	Forward	Joint Optim	DMI	T-revision	CDR	Ours
MNIST Sym-50%	97.51% ±0.28%	90.13% ±0.09%	91.68% ±0.21%	97.86% ±0.22%	97.79% ±0.13%	97.04% ±1.15%	98.38% ±0.21%	98.13% ±0.17%	**98.52%** ±0.09%
CIFAR10 Sym-50%	77.11% ±0.43%	70.71% ±0.24%	72.80% ±0.45%	77.92% ±0.66%	85.00% ±0.17%	78.28% ±0.48%	83.40% ±0.65%	82.64% ±0.89%	**86.40%** ±0.34%
CIFAR100 Sym-50%	39.73% ±2.74%	38.45% ±0.25%	51.60% ±0.49%	38.59% ±1.62%	57.97% ±0.67%	49.81% ±1.22%	57.71% ±0.84%	55.30% ±0.96%	**58.06%** ±0.59%

high, e.g., noise rate 80%, the number of extracted examples may become too small, which means there are not enough examples to sufficiently train the model. This could be addressed by making use of the non-confident examples by using the semi-supervised learning method, e.g., SELF [6], DivideMix [4]. However, our aim is to verify the effectiveness of the method to extract high-quality confident examples, not to boost the classification performance.

4. Me-Momentum with clean validation sets

We conduct experiments with clean validation sets. In Table 4, we can observe that a clean validation set will help to select a better model compared with the noisy validation set. The final performance will become better.

Table 4. Means and standard deviations of classification accuracy with clean validation set on *CIFAR10*

	with noisy validation set	with clean validation set
Sym-20	91.44 ± 0.33%	91.60 ± 0.31%
Sym-40	88.39 ± 0.34%	89.14 ± 0.51%
Sym-50	86.40 ± 0.34%	86.88 ± 0.70%
Inst-20	90.86 ± 0.21%	91.34 ± 0.24%
Inst-40	86.66 ± 0.91%	87.80 ± 0.84%

5. Comparison of training time

We compare the training time with representative baselines on *CIFAR10* with ResNet18 in Table 5. The number of the inner loop varies because the inner loop stops by exploiting a noisy validation set. Note that we discard non-confident examples and only use confident examples to train the model. The training time in each round would be much less than the normal training.

Table 5. Comparison of training time with different baselines on *CIFAR10*

Methods	Training time
CE	68 mins
JointOptim	88 mins
Co-teaching	91 mins
T-revision	232 mins
CDR	178 mins
Ours Sym-50	169.4 ± 18.8 mins
Ours Inst-40	160.0 ± 21.0 mins

Note that CE stands for the normal neural network training by employing the cross-entropy loss function. The total number of epochs for CE is set to 200. Since the number of the inner loop varies in the proposed method, we have reported the average training time of five runs and the standard deviation. Note that Me-momentum has a smaller training time on the instance-40% noise than that on the symmetric-50% noise. This may be caused by that the former setting is more difficult than the latter one and less confident examples are extracted. The conclusion is that the training time of the proposed method is shorter than T-revision and CDR but longer than Co-teaching, and JointOptim.

6. Experiments complementary to Clothing1M

Clothing1M contains 1 million examples with real-world noisy labels, the number of examples cross classes are significantly imbalanced. Moreover, the ratio of examples cross classes in the training dataset is also dramatically different from that of the validation and test dataset.

A common way to deal with the imbalance is to make the number of training examples in each class the same. That is to say, we need to reduce the number of training examples for different classes to the least number of the classes. Therefore, the method cannot fully use training examples, especially on an extremely imbalanced training dataset. We take two approaches to deal with it. 1) We introduce the importance reweighting technique to different examples [3]. Suppose the number of samples for class j on the training (validation) dataset is N^j_{train} (resp. N^j_{val}). The weight of samples in class j is calculated as:

$$W^j = \frac{\frac{1}{C}\sum_{j=1}^{C} N^j_{train}}{N^j_{train}} \times \frac{N^j_{val}}{\frac{1}{C}\sum_{j=1}^{C} N^j_{val}}, \qquad (1)$$

where C is the total number of classes.

2) We extract the confident examples separately according to the class. Specifically, we select a classifier based on the highest score in each class, where the score S for class j is obtained with:

$$S^j = \beta \times Accuracy^j + (1-\beta) \times Precision^j. \qquad (2)$$

We set β to 0.45 in experiments. Then, we use the classifier to extract the confident examples in this class.

7. Experiments complementary to Section 3.1

In Section 3.1, we discuss the statistics of the extracted confident examples. However, due to the space limit, in the paper, we only provide parts of the empirical results.

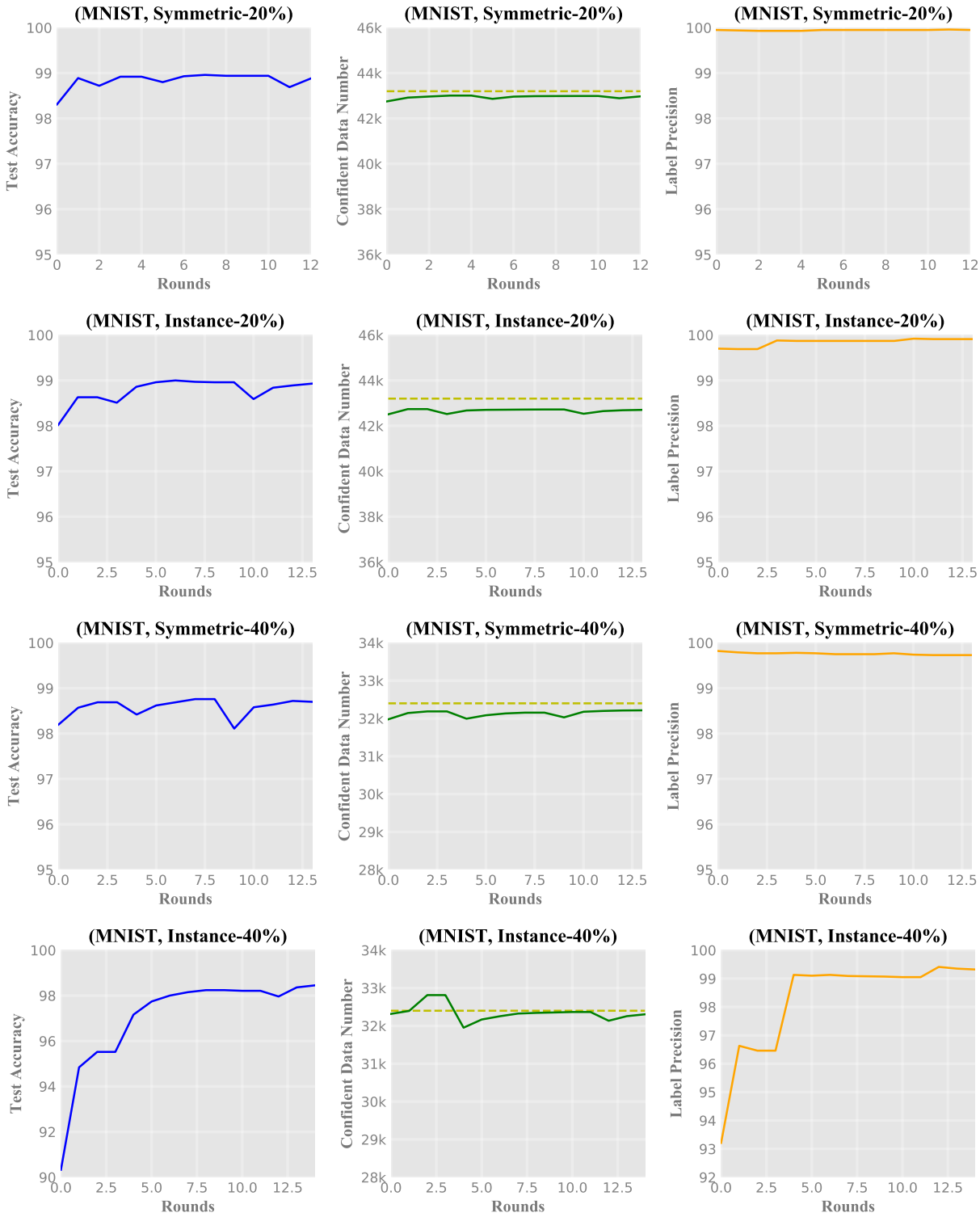

Figure 1. Statistics of the extracted confident examples on *MNIST* by Me-Momentum. We call one update of the classifier and the extracted confident examples as one round. We illustrate how the label precision of the extracted confident examples, the number of the extracted confident examples, and the classification accuracy of the classifier trained by using the extracted confident examples change during the training of Me-Momentum. The dash lines in the middle column indicate the number of clean labels in the noisy training data.

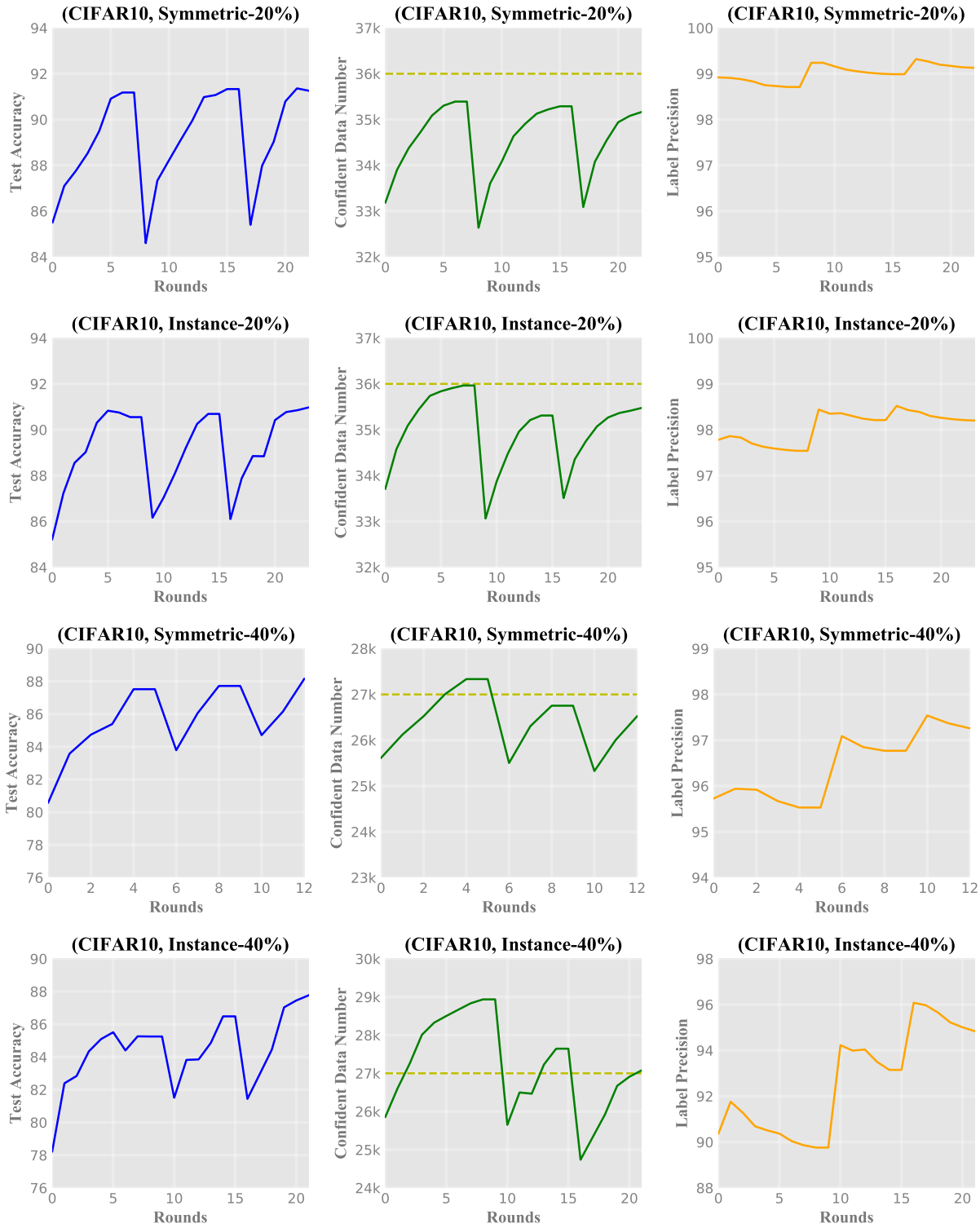

Figure 2. Statistics of the extracted confident examples on *CIFAR10* by Me-Momentum. We call one update of the classifier and the extracted confident examples as one round. We illustrate how the label precision of the extracted confident examples, the number of the extracted confident examples, and the classification accuracy of the classifier trained by using the extracted confident examples change during the training of Me-Momentum. We have three distinct peaks in these figures because we have set $N_{\text{outer}} = 3$ and the classifiers are re-initialized in the outer loop. The dash lines in the middle column indicate the number of clean labels in the noisy training data.

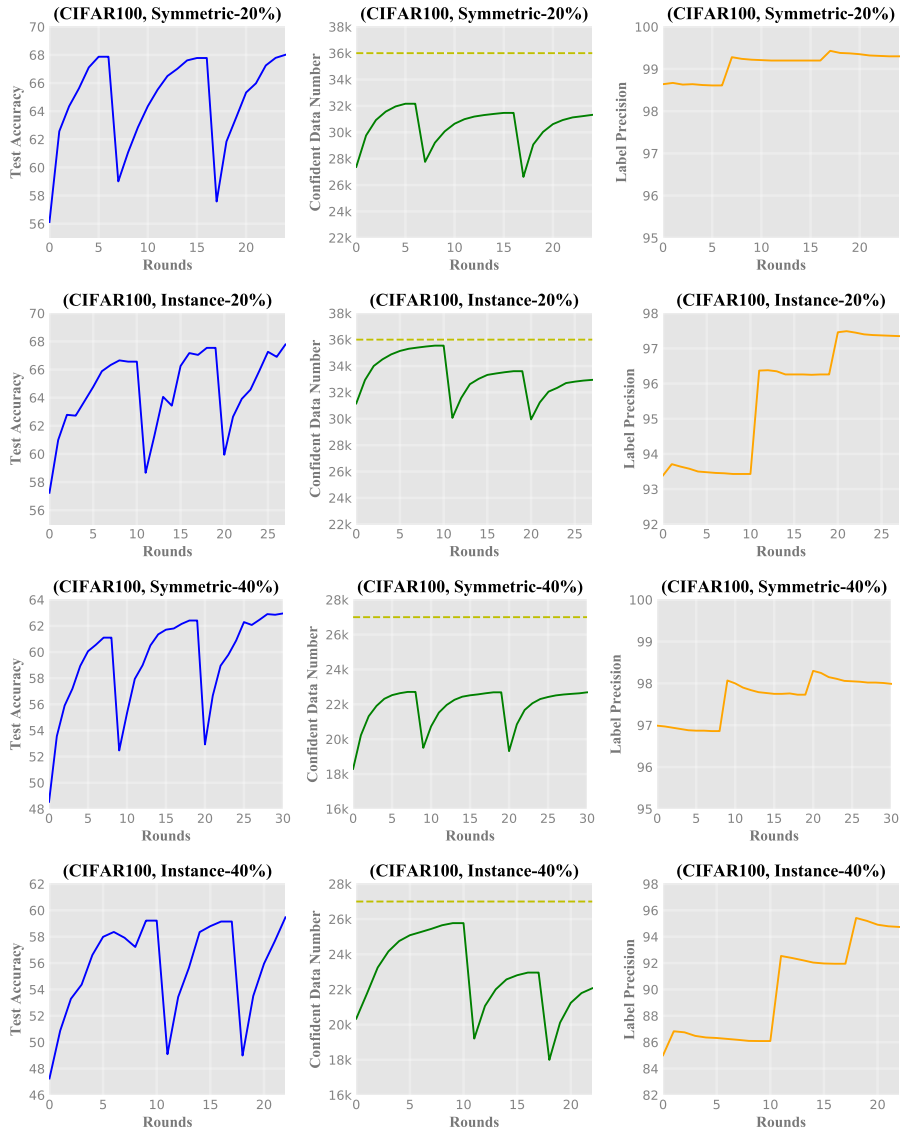

Figure 3. Statistics of the extracted confident examples on *CIFAR100* by Me-Momentum. We call one update of the classifier and the extracted confident examples as one round. We illustrate how the label precision of the extracted confident examples, the number of the extracted confident examples, and the classification accuracy of the classifier trained by using the extracted confident examples change during the training of Me-Momentum. We have three distinct peaks in these figures because we have set $N_{\text{outer}} = 3$ and the classifiers are re-initialized in the outer loop. The dash lines in the middle column indicate the number of clean labels in the noisy training data.

8. Experiments complementary to Section 3.2

In Section 3.2, we have visualized the extracted confident examples by using t-SNE [5]. It verifies that the proposed Me-Momentum is effective to extract hard confident examples. However, due to the space limit, in the paper, we only provide parts of the empirical results. In this supplementary material, we provide the visualization of all the employed datasets and settings.

Specifically, we show how the confident examples are progressively extracted in the inner and outer loops. In the figures, green, blue, and red dots represent confident examples extracted at the beginning, middle, and end rounds of the loops, respectively.

On the datasets of *MNIST* and *CIFAR10*, we can clearly see that the blue and red dots are mostly located at the boundaries of the clusters of green dots. On *CIFAR100*, we can also clearly see that there are lots of blue and red dots which are outside of the green clusters in the second and fourth figures. This supports and justifies our claim that Me-Momentum is able to extract hard confident examples (those are close to the decision boundary).

The figures are presented on the following pages.

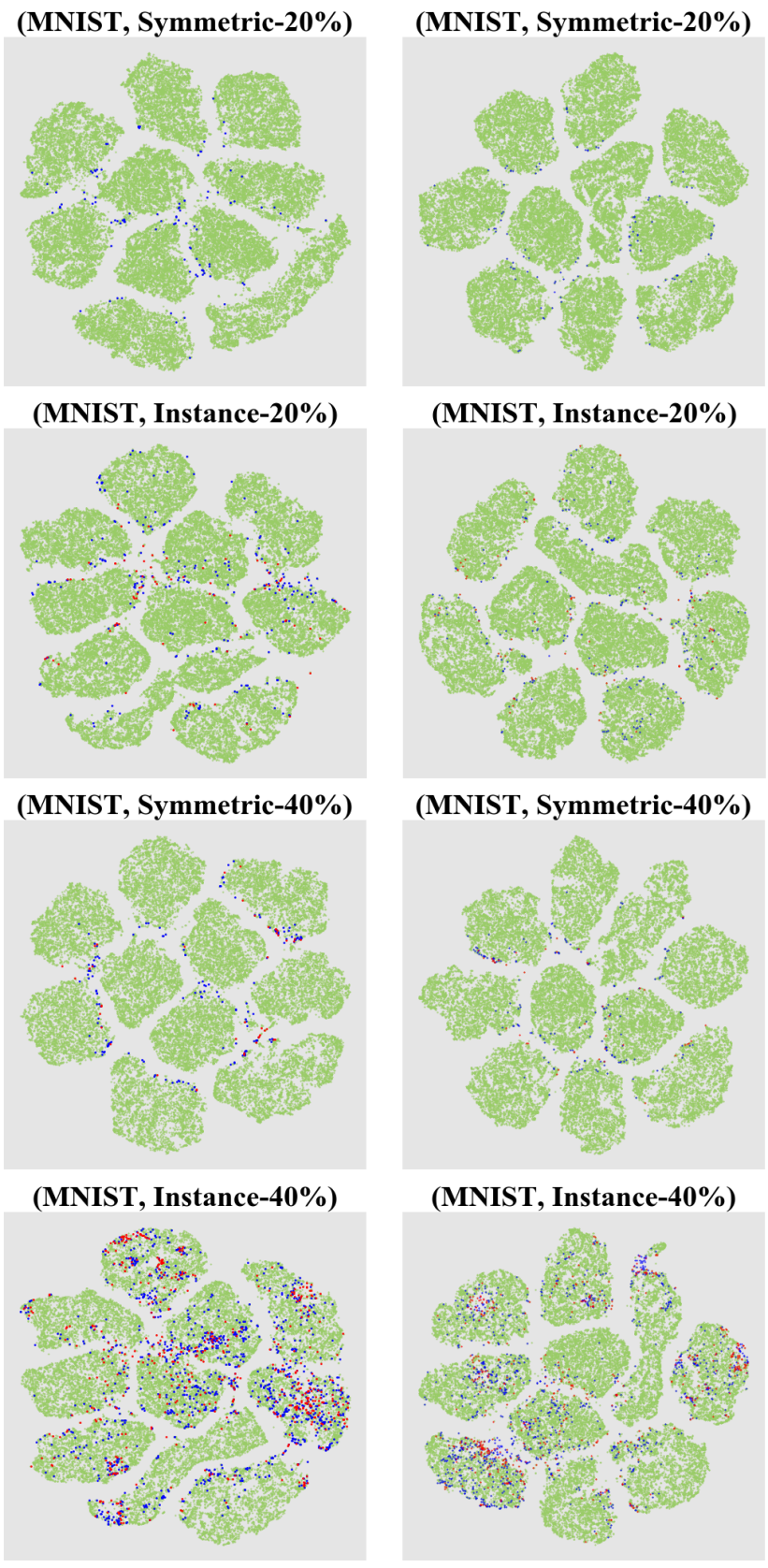

Figure 4. Visualization of the extracted confident examples on *MNIST*. The first column is about the confident data extracted in the first run of the inner loop; while the second column is about the confident data extracted in the outer loop.

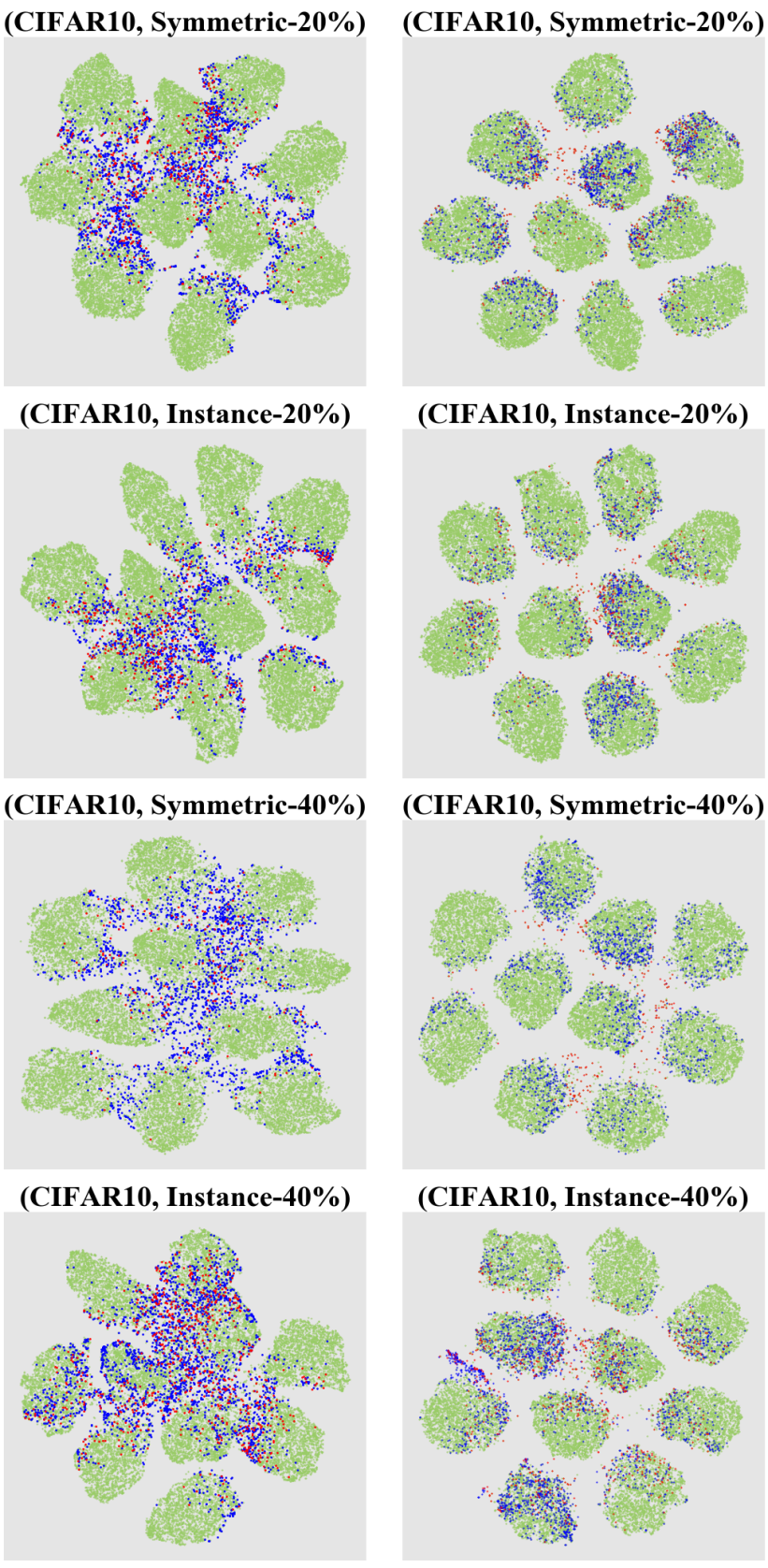

Figure 5. Visualization of the extracted confident examples on *CIFAR10*. The first column is about the confident data extracted in the first run of the inner loop; while the second column is about the confident data extracted in the outer loop.

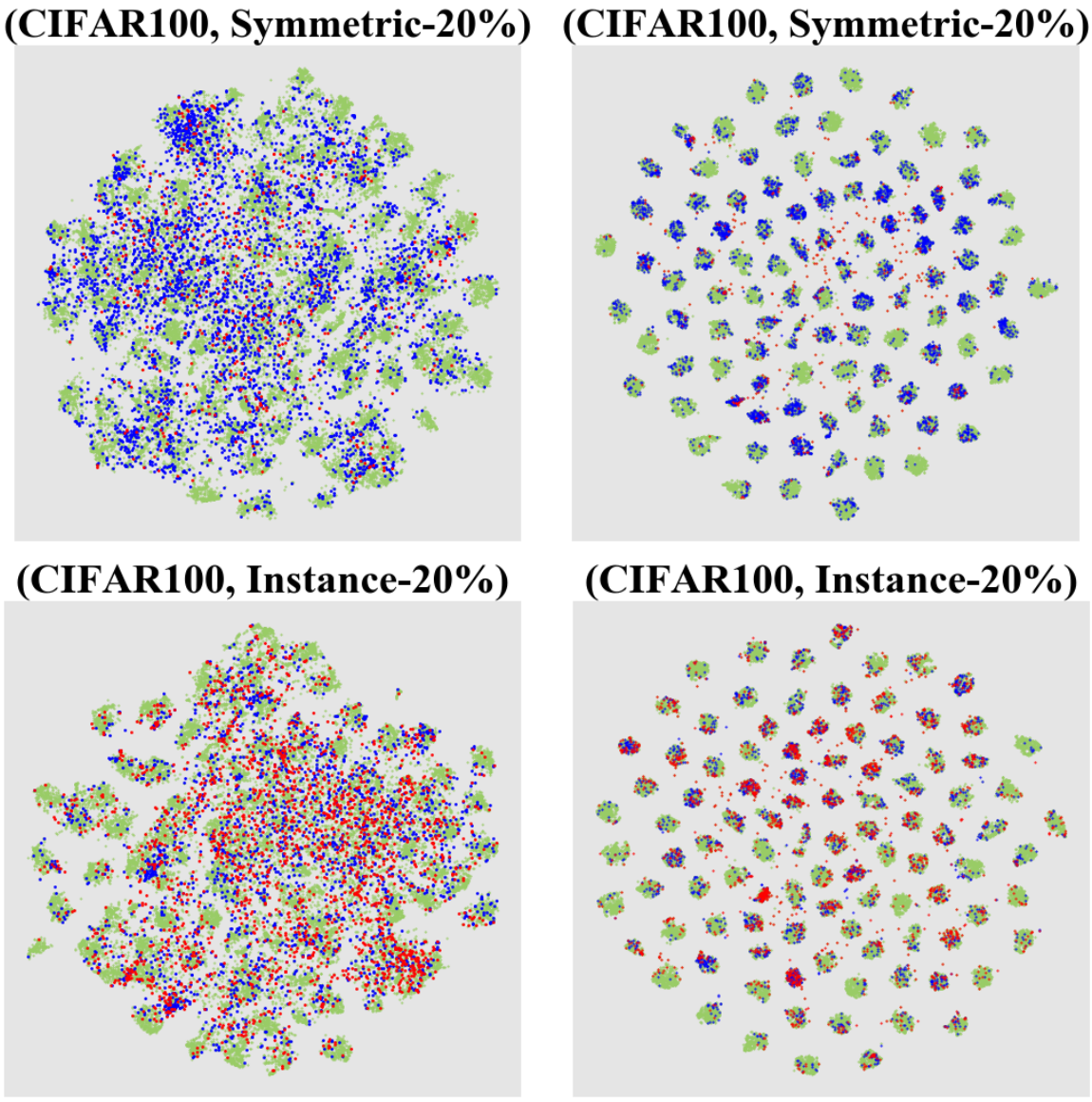

Figure 6. Visualization of the extracted confident examples on *CIFAR100*. The first column is about the confident data extracted in the first run of the inner loop; while the second column is about the confident data extracted in the outer loop.

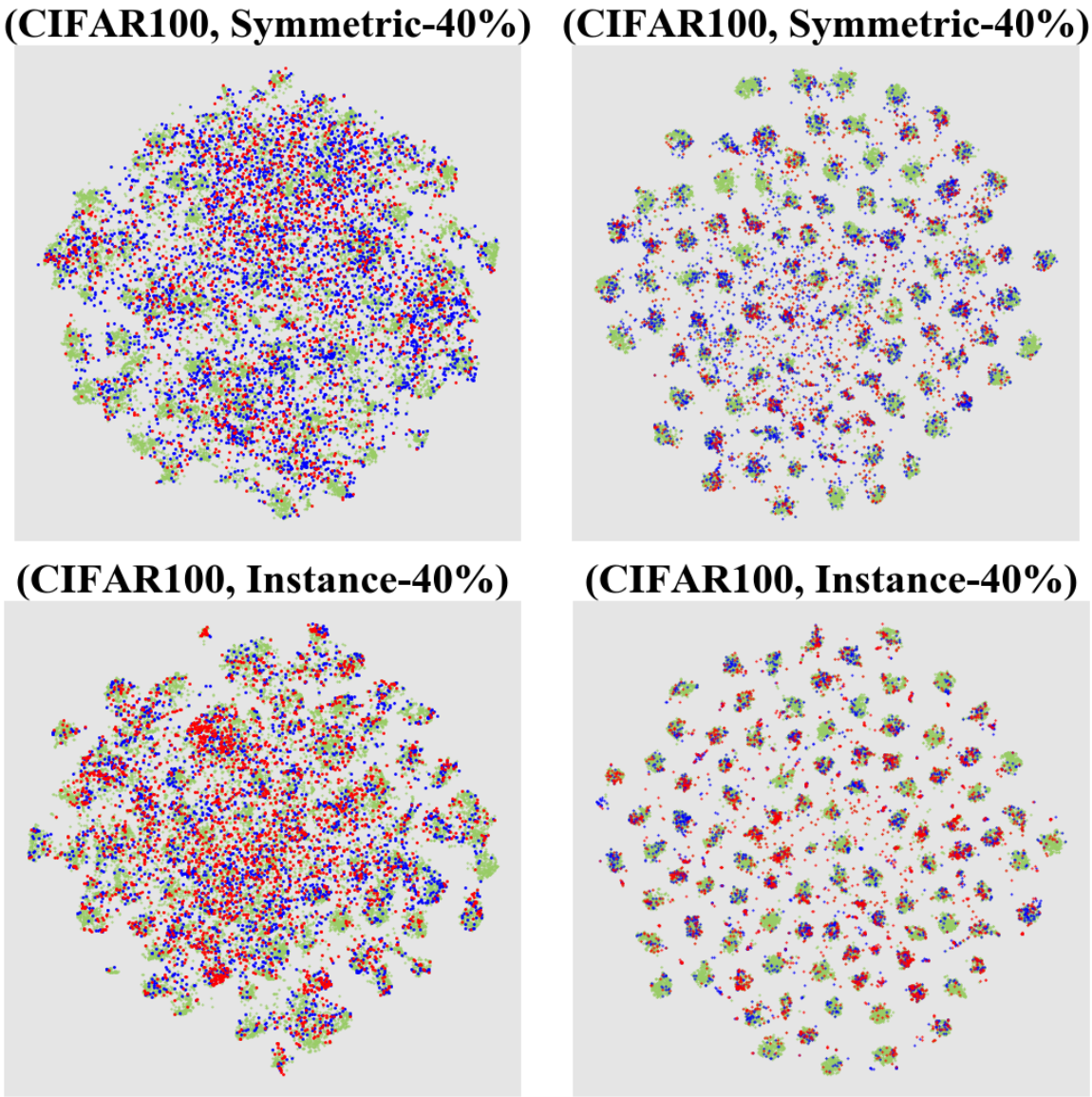

Figure 7. Visualization of the extracted confident examples on *CIFAR100*. The first column is about the confident data extracted in the first run of the inner loop; while the second column is about the confident data extracted in the outer loop.

References

[1] Xavier Gastaldi. Shake-shake regularization. *arXiv preprint arXiv:1705.07485*, 2017.

[2] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, pages 8527–8537, 2018.

[3] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 5375–5384. IEEE Computer Society, 2016.

[4] Junnan Li, Richard Socher, and Steven C. H. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *ICLR*, 2020.

[5] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[6] Tam Nguyen, C Mummadi, T Ngo, L Beggel, and Thomas Brox. SELF: learning to filter noisy labels with self-ensembling. In *ICLR*, 2020.

[7] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, pages 1195–1204, 2017.

[8] Xiaobo Xia, Tongliang Liu, Bo Han, Nannan Wang, Mingming Gong, Haifeng Liu, Gang Niu, Dacheng Tao, and Masashi Sugiyama. Parts-dependent label noise: Towards instance-dependent label noise. In *NeurIPS*, 2020.