Supplementary material for Addressing out-of-distribution label noise in webly-labelled data

Paul Albert, Diego Ortego, Eric Arazo, Noel E. O'Connor, Kevin McGuinness

School of Electronic Engineering,
Insight SFI Centre for Data Analytics, Dublin City University (DCU)
paul.albert@insight-centre.org

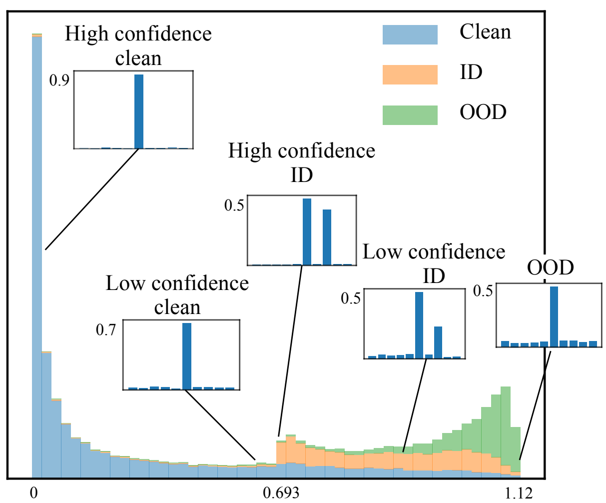

Figure 1: Clean/ID/OOD hierarchy established by the collision entropy of the interpolated label l_{detect}

1. Additional explanation of the behavior of the ID/OOD measure

Figure 1 illustrates the behavior of our proposed metric l_{detect}. By studying the collision entropy of the interpolated label y_{inter} between the network prediction and the ground truth label, we establish a hierarchy from clean to ID noise to OOD noise. The pivot point $-\log(.5) = 0.693$ marks the separation between low confidence clean samples and high confidence noisy samples. Although some clean samples will be detected as noisy at the pivot point, because we avoid OOD sample during this transition, we can correct the detected confident ID samples without concerns of labeling OOD data or corrupting the clean samples since we relabel correct but simply under-confident clean samples: this will not harm the training procedure (their label stays the same). By smoothing OOD samples, we also avoid correcting ID noisy samples with an under-confident corrected prediction.

2. Hyperparameter table

Table 1 details the hyperparameters used in every experiment reported in the state-of-the art comparison. The configuration remains the same across different noise ratios for miniImageNet and Stanford Cars. The parameters common to all experiments are: entropy regularization [2], SGD optimizer, a learning rate decay factor of 10, random horizontal flips, mixup [3] data augmentation. To match the baseline of [1], we add a dropout layer before the fully connected layer in the case of the Stanford Cars experiments. We do not use dropout for other datasets as we manage to match the baselines without it.

3. Algorithm

Alg. 1 displays the DSOS algorithm.

4. Examples of labeled images from the mini-Webvision subset

Figures 2 and 3 display examples of images labeled from the mini-Webvision subset. The annotations are available together with our code at [github].

References

[1] Lu Jiang, Di Huang, Mason Liu, and Weilong Yang. Beyond Synthetic Noise: Deep Learning on Controlled Noisy Labels. In *International Conference on Machine Learning (ICML)*, 2020.

[2] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa. Joint Optimization Framework for Learning with Noisy Labels. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[3] H. Zhang, M. Cisse, Y.N. Dauphin, and D. Lopez-Paz. mixup: Beyond Empirical Risk Minimization. In *International Conference on Learning Representations (ICLR)*, 2018.

Table 1: Hyperparameter variations across experiments. We do not change hyperparameters across noise levels for CIFAR-100, miniImageNet, and Stanford Cars.

	CIFAR-100	Stanford Cars	miniImageNet	Webvision	Clothing1M
Network	PreActResNet18	InceptionResNetV2	InceptionResNetV2	InceptionResNetV2	ResNet50
ImageNet pretraining	No	No	No	No	Yes
Number of epoch	100	400	200	100	100
Batch size	32	32	32	32	32
Initial learning rate	0.03	0.05	0.01	0.01	0.002
Lr reduction	$[50, 80]$	$[200, 300]$	$[100, 160]$	$[50, 80]$	$[50, 80]$
Weight decay	$5e-4$	$5e-4$	$5e-4$	$5e-4$	$1e-3$
Resize	32	320	320	256	256
RandomResize Range	–	$[0.75, 1.33]$	–	–	–
Crop	32	299	299	227	224
Dropout ratio	0.0	0.3	0.0	0.0	0.0
α	0.05	0.05	0.05	0.05	0.05
Epoch start correction	51	201	101	51	1

Algorithm 1 DSOS

Input: $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ a web noise dataset. h at convolutional neural network.
Parameters: $\alpha, e_{warmup}, e_{max}$
Output: Trained neural network h_ϕ

for $e = 1, \dots e_{warmup}$ **do** ▷ Warmup
  for $t = 1, \dots numBatches$ **do**
    Sample the next mini-batch (x, y) from $\mathcal{D}$
    $L = CrossEntro(h(x_{mixed}), y_{mixed})$
    $UpdateNetworkWeights(\mathrm{L})$
  end for
end for
for $e = e_{warmup} + 1, \dots e_{max}$ **do** ▷ Label correction
  $\tilde{U}, \tilde{V}, predictions = EvaluateMetrics(h, \mathcal{D})$ ▷ Evaluated with regards to the original labels
  for $y_i = y_1, \dots y_N$ **do**
    if $\tilde{u}_i > 0.9$ **then** ▷ In-distribution bootstrapping, $\tilde{U} = \{\tilde{u}_i\}_{i=1}^{N}$
      $y_i = p_i$ ▷ $predicitions = \{p_i\}_{i=1}^{N}$
    end if
    $y_i = Softmax(y_i v_i / \alpha)$ ▷ Dynamic Softening, $\tilde{V} = \{\tilde{v}_i\}_{i=1}^{N}$
  end for
  for $t = 1, \dots numBatches$ **do**
    Sample the next mini-batch (x, y) from $\mathcal{D}$ ▷ Train on the corrected labels
    $\tilde{V}_{mini}$ the values in $\tilde{V}$ for the samples in the mini-batch
    $L = CrossEntro(h(x), y)$
    $L = L + 0.4 \times EntroPen(h(x), \tilde{V}_{mini})$ ▷ Weighted entropy penalization
    $UpdateNetworkWeights(\mathrm{L})$
  end for
end for
return h ▷ Robustly trained network

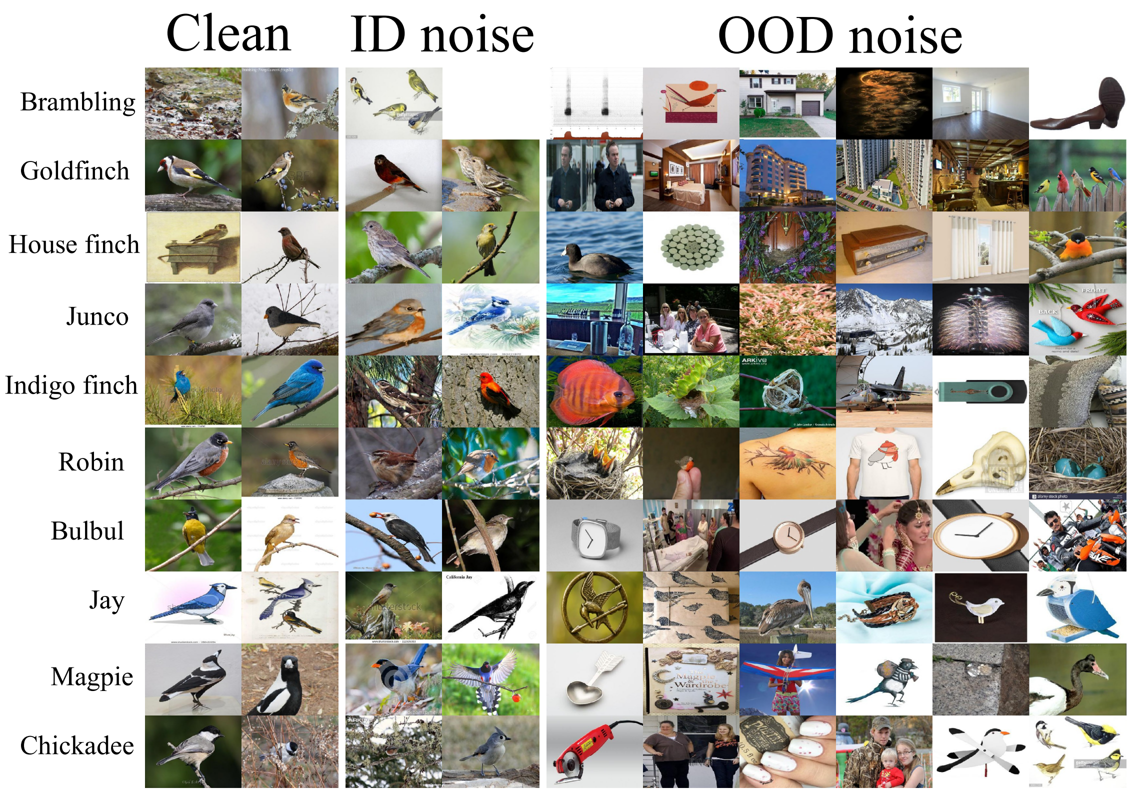

Figure 2: Samples annotated as clean, in-distribution noise, out-of-distribution noise.

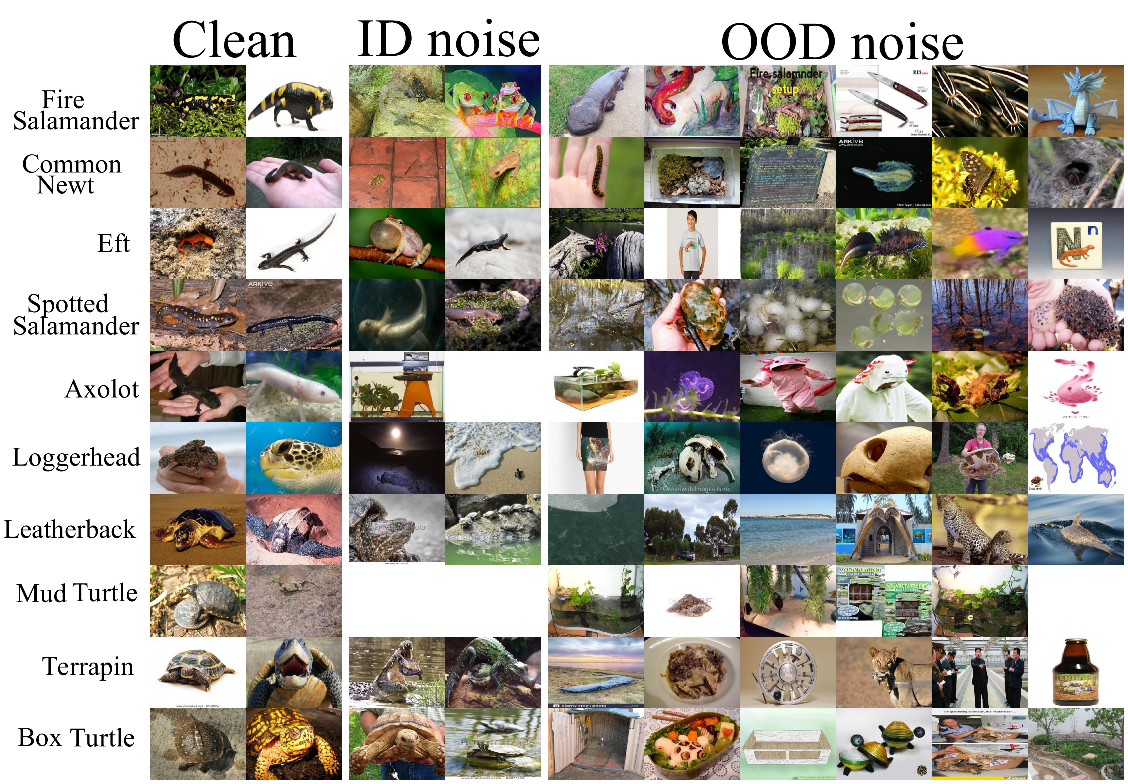

Figure 3: Samples annotated as clean, in-distribution noise, out-of-distribution noise.