Towards Robust Learning with Different Label Noise Distributions

Diego Ortego, Eric Arazo, Paul Albert, Noel E. O'Connor and Kevin McGuinness
Insight Centre for Data Analytics, Dublin City University (DCU)
{diego.ortego, eric.arazo}@insight-centre.org

Abstract

Noisy labels are an unavoidable consequence of automatic image labeling processes to reduce human supervision. Training in these conditions leads Convolutional Neural Networks to memorize label noise and degrade performance. Noisy labels are therefore dispensable, while image content can be exploited in a semi-supervised learning (SSL) setup. Handling label noise then becomes a label noise detection task. Noisy/clean samples are usually identified using the small loss trick, which is based on the observation that clean samples represent easier patterns and, therefore, exhibit a lower loss. However, we show that different noise distributions make the application of this trick less straightforward. We propose to continuously relabel all images to reveal a loss that facilitates the use of the small loss trick with different noise distributions. SSL is then applied twice, once to improve the clean-noisy detection and again for training the final model. We design an experimental setup for better understanding the consequences of differing label noise distributions and find that non-uniform out-of-distribution noise better resembles real-world noise. We show that SSL outperforms other alternatives when using oracles and demonstrate substantial improvements across five datasets of our label noise Distribution Robust Pseudo-Labeling (DRPL). We further study the effects of label noise memorization via linear probes and find that in most cases intermediate features are not affected by label noise corruption. Code and details to reproduce our framework will be made available.

1. Introduction

Modern representation learning, i.e. the extraction of useful information to build classifiers or other predictors [5], in computer vision is led by Convolutional Neural Networks (CNNs) [9, 28, 4, 31, 49, 25, 11]. Their widespread use is attributable to their capability to model complex patterns when vast amounts of labeled data are available [24]. This supervision requirement limits exploiting the vast amounts of web images as it is infeasible to label every image for each particular task. However, leveraging these data might drive visual representation learning a step forward.

What can we do to relax supervision? One could adopt transfer learning or domain adaptation [39], where representations from a source domain are transferred to a target domain where fewer labels are available. This approach, however, makes the target domain dependent on the source one. Learning representations from scratch in the target domain, on the other hand, may lead to better representations. Several alternatives exist: semi-supervised learning (SSL), which jointly learns from few labeled images and extensive unlabeled ones [6, 1]; self-supervised learning, where data provides the supervision [29, 24]; or learning with label noise, where automatic labeling processes introduce noise in the observed labels [36, 44]. This paper focuses on this last alternative, which has attracted much recent interest [43, 44, 2, 42].

Learning with label noise is challenging; recent studies on the generalization capabilities of deep networks [45] demonstrate that noisy labels are easily fit by CNNs, harming generalization. There is, however, a key observation on how CNNs memorize corruptions: they tend to learn easy patterns first and these patterns are closer to clean data patterns, i.e. correctly labeled images [45, 3], thus exhibiting lower loss than images with noisy or incorrect labels. This phenomenon is commonly named small loss [36, 44, 43] and recent works exploit this *small loss trick* to identify clean and noisy samples [14, 34, 2].

Approaches dealing with label noise can be categorized into: loss correction [32, 30, 22, 2], when the loss is weighted to correct the label noise effect; relabeling [36, 44], when all observed labels are corrected by an estimation of the true labels; and approaches that discard noisy labels to transform the problem into SSL [10, 23]. Despite the variety of approaches and comparative evaluations, it is not clear which of them behave better. Most approaches are exhaustively compared on CIFAR data [26] and then tested in a real world datasets such as WebVision [27]. Although substantial improvements have been achieved in recent years when training from scratch on CIFAR [23], transfer learning is usually used when approaches are evaluated on WebVision [22, 43, 7]. This misalignment might harm the ability to understand rep-

resentation learning from scratch as it has been shown that fine-tuning pre-trained weights is robust to label noise [18].

In light of these limitations, we undertake an exhaustive study on different label noise distributions and dataset complexities and propose a general solution to tackle all of them. In particular, we adopt the 32×32 and 64×64 versions of the ImageNet dataset and artificially introduce 4 different label noise types (2 in-distribution and 2 out-of-distribution). We find that discarding noisy labels and training with a semi-supervised technique outperforms all other approaches considered. The novelty here lies in the clean-noisy detection strategy, which is based on both relabeling [36] and the small loss trick [2]. Relabeling modifies the dataset noise separating the loss of clean and noisy samples to facilitate the use of the small loss trick [2]. SSL via pseudo-labeling [1] is then applied twice to refine the clean-noisy detection before a final training stage. Our contributions include:

- A framework to study label noise with different distributions providing a more realistic understanding.
- A label noise detection method that substantially outperforms other recent methods [10, 23, 34] and consistently works for different label noise distributions.
- An experimental demonstration, based on using oracles, that SSL substantially outperforms loss correction based on label noise transition matrices [30].
- A study of label noise memorization and its importance for representation learning, showing that despite the performance degradation for source and target tasks due to the label noise fitting, the intermediate representations do not often suffer such degradation.
- An extensive evaluation in five datasets using multiple label noise distributions and training from scratch, demonstrating both superior performance of our label noise Distribution Robust Pseudo-Labeling (DRPL) approach and contributing to a better understanding of existing methods.

2. Related work

Label noise is a well-known problem in machine learning [12]. Recent efforts in image classification focus on dealing with in-distribution noise [23], where the set of possible labels S is known and noisy labels belong to this set. However, label noise might also come from outside the distribution [40], which occurs in many real-world scenarios [27].

Several label noise distributions can affect dataset annotations, namely *uniform* or *non-uniform* random label noise. Uniform label noise means the true labels are flipped to a different class with uniform random probability. Non-uniform noise has different flipping probabilities for each class.

Loss correction approaches [32, 30, 2, 43] either modify the loss directly or the network probabilities to compensate for the incorrect guidance provided by the noisy samples. [32] extend the loss with a perceptual term that introduces a reliance on the model prediction. Han et al. [15] adopt the same approach, but differ in the estimation of the perceptual term as network predictions are replaced by class estimations based on class prototypes. These approaches are, however, limited in that the noisy label always affects the objective. Arazo et al. [2] address this by dynamically weighting the original and perceptual terms based on clean-noisy per-sample probabilities given by a label noise model. Patrini et al. [30] estimate the label noise transition matrix T, which specifies the probability of one label being flipped to another, and correct the softmax probability by multiplying by T. In the same spirit, Yao et al. [43] propose to estimate T in a Bayesian non-parametric form and deduce a dynamic label regression method to iteratively infer the latent true labels and jointly train the classifier and model the noise.

Other loss correction approaches reduce the contribution of noisy samples to the loss. [22] propose to use a mentor network that learns a curriculum (i.e. a weight for each sample) to guide a student network that learns under noise conditions. Similarly, [13] learn a curriculum based on an unsupervised estimation of data complexity. Wang et al. [40] introduce a weighting scheme in the loss to reduce the contribution of noisy samples, and a metric learning framework to pull noisy samples representations away from those of clean ones. Robust loss functions are studied in [48], which define the generalized cross-entropy loss by jointly exploiting the benefits of mean absolute error and cross-entropy losses. [42] propose the Determinant-based Mutual Information, a generalized version of mutual information that is provably insensitive to noise patterns and amounts.

Other approaches relabel the noisy samples by modeling their noise through conditional random fields [37], or CNNs [38] using a small set of clean samples, which limits their applicability. Tanaka et al. [36] have, however, demonstrated that it is possible to do sample relabeling using the network predictions as soft labels. [44] further improve [36] by using estimated label distributions as soft pseudo-labels.

A simple approach to dealing with label noise is to remove the corrupted data. This is not only challenging because difficult clean samples may be confused with noisy ones [40], but also removes the possibility of exploiting the noisy samples for representation learning. It has, however, recently been demonstrated [10, 23] that it is useful to discard samples that are likely to be noisy while still using them in a semi-supervised setup. [10] define clean samples as those whose prediction agrees with the label with a high certainty, whereas [23] use high softmax probabilities to distinguish clean samples after performing negative learning, i.e. minimizing the probability of predicting a class when the label

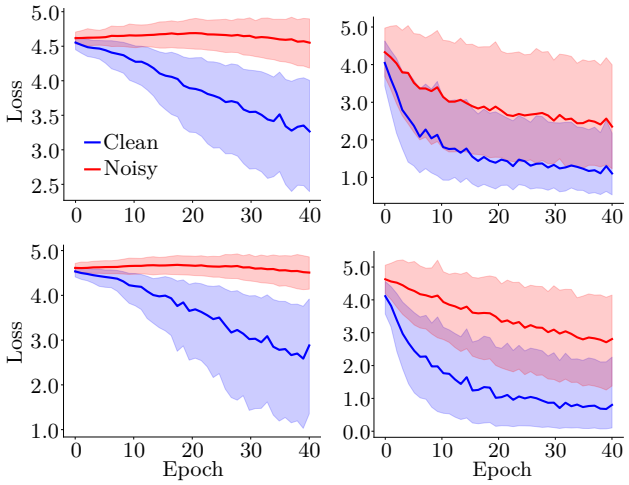

Figure 1. Loss for clean (blue) and noisy (red) samples for different label noise distributions in 100 classes of ImageNet32. 80% uniform in-distribution noise (top-left); 50% non-uniform in-distribution noise (top-right); 80% uniform out-of-distribution noise (bottom-left); and 50% non-uniform out-of-distribution noise (bottom-right). Training: 40 epochs with a PreAct ResNet-18 [17] with learning rate of 0.1 and cross-entropy loss.

comes from a random label flip that is likely incorrect.

3. From label noise to semi-supervised learning

Image classification can be formulated as learning a model $h_\theta(x)$ from a set of training examples $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ with $y_i \in \{0,1\}^C$ being the one-hot encoding true label corresponding to x_i. In our case, h_θ is a CNN and θ represents the model parameters (weights and biases). As we are considering classification under label noise, the label y_i can be noisy (i.e. x_i is a noisy sample). This training under label noise can be redefined through SSL, where the N samples are split into a set of N_u unlabeled samples $\mathcal{D}_u = \{x_i\}_{i=1}^{N_u}$ and a set of N_l labeled samples $\mathcal{D}_l = \{(x_i, y_i)\}_{i=1}^{N_l}$. Ideally, $\mathcal{D}_l$ would contain the clean samples and $\mathcal{D}_u$ the noisy ones. In practice, clean and noisy samples sets are unknown and must be estimated. Detected clean (noisy) samples are used as the labeled (unlabeled) set and, as such, detection accuracy will influence the SSL success. The remainder of this section introduces the proposed two-stage label noise detection (Subsections 3.1 and 3.2) and briefly explains the SSL approach [1] (Subsection 3.3).

3.1. Label noise detection: first stage

Recent literature assumes that using the small loss trick when training with cross-entropy leads to accurate clean-noisy discrimination [14, 34, 2]. However, as Figure 1 shows, different label noise distributions result in different behavior. Noisy samples tend to have higher loss than clean samples for both in-distribution and out-of-distribution noise but the different noise types exhibit different complexities. The uniform noise types (Figure 1 (left)) exhibit higher separation between the losses of clean (blue) and noisy (red) samples than the non-uniform ones (Figure 1 (right)). This shows that a straightforward application of the small loss trick will likely encounter some difficulties.

Based on the evidence that clean data is easier to fit than mislabelled data [3], we propose to identify the clean data by fully relabeling all samples using the network predictions and analyzing which samples still fit the original labels. The relabeling approach optimizes the following loss function:

$$\ell_t(\phi) = \begin{cases} -\frac{1}{N}\sum_{i=1}^{N} y_i^T \log\left(h_\phi(x_i)\right) + R & t \leq q \\ -\frac{1}{N}\sum_{i=1}^{N} \tilde{y}_i^T \log\left(h_\phi(x_i)\right) + R & t > q \end{cases}, \quad (1)$$

$$R = \lambda_1 R_H + \lambda_2 R_A, \quad (2)$$

where t indexes the epochs, q is a constant that defines the number of warm-up epochs with the original (potentially noisy) labels y, and R_H and R_A are two regularization terms (see details in [36]) weighted by λ_1 and λ_2 included to ensure convergence. The first phase ($t \leq q$) trains without relabeling and uses a high learning rate to prevent fitting the label noise, while the second ($t > q$) relabels all samples by computing soft pseudo-labels $\tilde{y}$ that are re-estimated every epoch using the softmax predictions $h_\phi(x)$. For simplicity we omit the epoch index t inside Eq. (1).

The new labels or pseudo-labels $\tilde{y}$ no longer represent the original label noise, but the noise from inaccurate network predictions. Figure 2 illustrates the benefits of this approach on the detection of clean and noisy samples. The relabeling approach progressively fits the new labels (Figure 2 (a)) as it is highly affected by confirmation bias [1], i.e. overfitting to incorrect pseudo-labels predicted by the network. However, this reveals an interesting property in the cross-entropy loss $\ell^*(\phi) = -\frac{1}{N}\sum_{i=1}^{N} y_i^T \log\left(h_\phi(x_i)\right)$ with respect to the original labels y (Figure 2 (b)). Clean samples tend to agree with the original labels (blue loss is low) substantially better than noisy samples (red loss is high). This facilitates distinguishing between clean and noisy samples using the loss, which does not occur in a standard training (Figure 1). Given the loss $\ell^*(\phi)$ ((*) denotes losses with respect the original labels y), we adopt a probabilistic version of the small loss trick [2]. In particular, the authors in [2] fit a 2-component Beta Mixture Model (BMM) to the loss to model the loss of clean (noisy) samples using the first (second) component, as lower losses correspond to clean samples. The probability of each sample being clean or noisy is then estimated using the posterior probability $p(k \mid \ell)$ from the mixture model. We use this BMM approach on the loss $\ell^*(\phi)$ to detect clean and noisy samples, thus defining the initial labeled and unlabeled sets as:

$$\mathcal{D}_l' = \{(x_i, y_i) : p(k=2 \mid \ell_i^*(\phi)) \leq \gamma_1\}, \quad (3)$$

$$\mathcal{D}_u' = \{(x_i) : p(k=2 \mid \ell_i^*(\phi)) > \gamma_1\}, \quad (4)$$

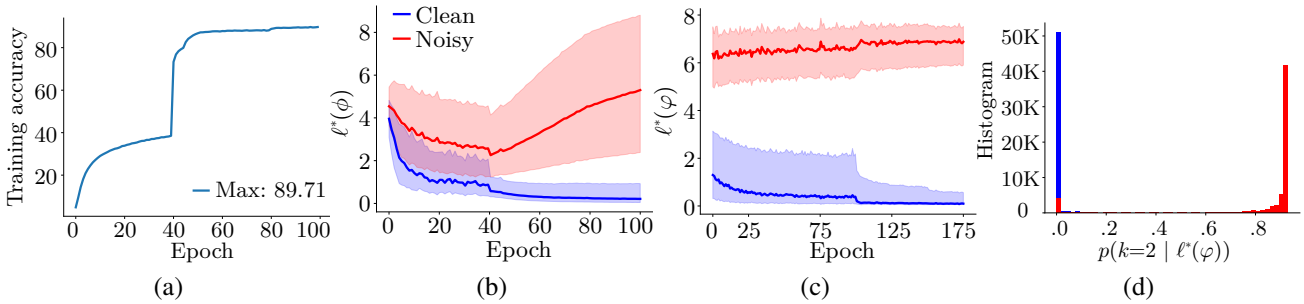

Figure 2. Example of proposed approach for label noise detection in ImageNet32. (a) Training accuracy for model $h_\phi(x)$ is high due to memorization of pseudo-labels. (b) Loss with respect to the original labels $\ell^*(\phi)$ shows that clean samples (blue) still fit the original labels y, while noisy samples (red) do not. (c) Loss $\ell^*(\varphi)$ for the model $h_\varphi(x)$ trained in a semi-supervised manner, where clean and noisy samples are further separated. (d) Posterior probability $p(k=2 \mid \ell^*(\varphi))$ obtained by the BMM used to generate $\mathcal{D}_l$ and $\mathcal{D}_u$. Figures correspond to 50% of non-uniform in-distribution noise (see Subsection 3).

where $k=2$ represents noisy samples, $\ell_i^*(\phi)$ is the loss after the relabeling approach for sample x_i, and γ_1 is a threshold to detect clean and noisy samples. This threshold should be small to select only highly probable clean samples (we use $\gamma_1 = 0.05$ unless otherwise stated).

3.2. Label noise detection: second stage

The second stage of our clean-noisy samples detector refines the first stage estimation by training a new semi-supervised model $h_\varphi(x)$ using the labeled and unlabeled sets from Eqs. (3) and (4). This new model $h_\varphi(x)$ leads to a loss $\ell^*(\varphi)$ with respect the original labels y that further facilitates the clean-noisy detection as training is performed with far less corrupted labels. Again, we detect clean and noisy samples by fitting a BMM to the loss $\ell^*(\varphi)$ and compute the final labeled and unlabeled sets $\mathcal{D}_l$ and $\mathcal{D}_u$ by selecting samples similarly to Eqs. (3) and (4). As Figure 2 (c) shows, clean and noisy samples become easily separable in the loss $\ell^*(\varphi)$ leading to wider separation in posterior probabilities $p(k \mid \ell^*(\varphi))$ (Figure 2 (d)). This allows the use of a higher threshold compared to γ_1 and lower risk of introducing noisy samples. We therefore apply a maximum a posteriori threshold over $p(k \mid \ell^*(\varphi))$ of $\gamma_2 = 0.5$. Subsection 5.2 demonstrates the capabilities of our label noise detection method for different label noise distributions.

3.3. The semi-supervised learning approach

We adopt a recent SSL approach described in [1] for simplicity and performance. This approach performs relabeling or pseudo-labeling as presented in the first stage, but the pseudo-labels are only estimated for the unlabeled samples. Mixup data augmentation [46] and label noise regularization are also applied to alleviate confirmation bias and make pseudo-labeling effective. The SSL approach is applied twice. The first time SSL is used in the second stage of the label noise detection, i.e. with the labeled $\mathcal{D}'_l$ and unlabeled $\mathcal{D}'_u$ sets (see Subsection 3.1) to train $h_\varphi(x)$, whereas the second time it is applied using the final labeled and unlabeled sets $\mathcal{D}_l$ and $\mathcal{D}_u$ (see Subsection 3.2) to train the final label-noise robust model $h_\theta(x)$. The supplementary material provides a summary of the proposed method.

4. Experimental setup

CIFAR data [26] is commonly corrupted for fast experimentation with label noise, whereas real-world performance against label noise is evaluated in datasets like WebVision [27]. To the best of our knowledge, however, the related work obviates the differences that might exist between artificially introduced noise and real-world noise. This section introduces the proposed label noise framework aimed at a better understanding of label noise (Subsection 4.1) and further describes other datasets commonly used for evaluation (Subsections 4.2 and 4.3). We use a PreAct ResNet-18 [17] in ImageNet32/64 and CIFAR and a ResNet-18 [16] for WebVision. We always train from scratch (except for the transfer learning experiments in Subsection 5.4), using SGD with a momentum of 0.9, a weight decay of 10^{-4}, batch size of 128 and initial learning rate of 0.1 for every stage of our method. Note that we do not use validation sets in any experiment. We take this decision due to the difficulty of defining a clean validation set in a real-world scenario and, more importantly, due to the fact that having clean data allows direct application of SSL, which leads to superior performance [1]. The supplementary material provides further experimental details.

4.1. ImageNet32/64

We propose to use ImageNet32/64 for fast experimentation and higher flexibility in better understanding label noise. ImageNet32/64 are 32×32 and 64×64 downsampled versions of the well-known ImageNet classification dataset [8]. This dataset contains 1.2M images uniformly distributed over 1000 classes. To introduce label noise we split the dataset into M in-distribution (ID) classes and $1000 - M$ out-of-distribution (OOD) classes. The split is performed to

study both ID noise, as is typically done in the literature [44], and also the less frequently considered OOD noise [40]. We set $M = 100$ (randomly selected classes) in all our experiments, thus leading to 127K images. We study both uniform and non-uniform noise in both ID and OOD scenarios. To introduce uniform noise for ID we randomly flip the true label to another of the M labels using uniform probabilities and excluding the true label, whereas for OOD we randomly select a class among the $1000 - M$ OOD classes and use an image to replace the ID image. To introduce non-uniform noise we use a label noise transition matrix [30] designed to be as realistic as possible. To this end, we average and apply row-wise unit-based normalization to the confusion matrices of the pre-trained ImageNet networks VGG-16 [33], ResNet-50 [16], Inception-v3 [35], and DenseNet-161 [20]. We truncate this 1000×1000 matrix and re-normalize it to the M classes for ID noise and the $1000 - M$ classes for OOD noise. We follow the same process as the uniform case to introduce noise, but using the row distributions corresponding to the true label of each image: we randomly flip the label for ID noise, while changing the image content for OOD noise. For a specific noise level r, we always keep $1 - r$ clean samples in each class and modify the remainder.

We use standard data augmentation by random horizontal flips and random 4 (8) pixel translations for ImageNet32 (64) in training. During the first stage of the label noise detection, we train for 40 epochs before starting 60 epochs of relabeling and reduce the learning rate by a factor of 10 in epochs 45 and 80. In the second stage of the label noise detection, we train 175 epochs and reduce the learning rate in epochs 100 and 150. The final SSL stage has 300 epochs with learning rate reductions in epochs 150 and 225.

4.2. CIFAR-10/100

The CIFAR-10/100 datasets [26] have 10/100 classes of 32×32 images split into 50K images for training and 10K for testing. We follow the criteria in [44] for label noise addition. For uniform noise, labels are randomly assigned excluding the original label. For non-uniform noise, labels are flipped with probability r to similar classes in CIFAR-10 (i.e. truck $\rightarrow$ automobile, bird $\rightarrow$ airplane, deer $\rightarrow$ horse, cat $\rightarrow$ dog), whereas for CIFAR-100 label flips are done to the next class circularly within the super-classes. Standard data augmentation by random horizontal flips and random 4 pixel translations is used in training. We train as in ImageNet32/64 in CIFAR-100, whereas in CIFAR-10 we slightly modify the first stage by training 130 epochs (70 before relabeling) and reduce the learning rate in epochs 75 and 110. This increase in training epochs is to ensure a better model before relabeling, as classes in CIFAR-10 are more different than in ImageNet32/64 and CIFAR-100, thus incorrect predictions during relabeling are less informative. We use $\gamma_1 = 0.1$ to assure sufficient labeled samples for SSL.

Table 1. ImageNet32/64 accuracy using oracles.

	Oracle SSL	Oracle forward
Uniform ID (80%)	**69.06 / 78.02**	34.66 / 49.98
Non-uniform ID (50%)	**73.24 / 81.80**	65.70 / 73.44

4.3. WebVision

We use WebVision 1.0 [27] to evaluate performance on real world label noise. We evaluate our approach using only the first 50 classes of WebVision as done in [7] resulting in a dataset of 137K images with resolution 224×224 after resizing and cropping. However, unlike most approaches in the literature [22, 7], we train all compared methods from scratch to better understand the effect of label noise. We use random horizontal flips during training and resize images to 256×256 before taking random 224×224 crops. For the first stage of the label noise detection, we train 40 epochs before starting 60 epochs of relabeling and reduce the learning rate dividing by 10 twice (epochs 45 and 80). For the second stage, we train 150 epochs and reduce the learning rate in epochs 100 and 125. The final SSL stage has 200 epochs with learning rate reductions in epochs 150 and 175.

5. Experiments

5.1. SSL vs label noise transition matrix

Correcting the loss by using the label noise transition matrix as proposed in [30] has recently attracted a lot of interest [19, 43, 41]. Estimating the label noise transition matrix T, however, is a challenging task as label flips from one class to another have to be estimated. It seems simpler, on the other hand, to detect clean and noisy samples and discard the noisy labels. Table 1 presents a study using oracles for both tasks (i.e. perfect knowledge clean-noisy samples and known T) to shed light the potential of each approach under ideal conditions. The results show that SSL surpasses label noise transition matrix correction [30] for both uniform and non-uniform noise. We believe this is an important finding that suggests further research on making the label noise transition matrix methods more effective.

5.2. Label noise detection comparison

Transforming the supervised training with label noise into SSL requires detecting the noise to, ideally, discard the labels and turn noisy samples into unlabeled ones. As commented in Section 1, many approaches use the small loss trick, i.e. considering low loss samples as clean ones, to accomplish such detection. However, Figure 1 shows that different noise distributions present different challenges, limiting a straightforward application of this trick. We confirm this limitation in Figure 3, where we compare, for different label noise distributions, the Receiver Operating Characteristic (ROC)

Table 2. Label noise detection results in ImageNet32. Key: NU (non-uniform); U (uniform); ID (in-distribution); OOD (out-of-distribution).

		NU-ID			U-ID			NU-OOD			U-OOD			
		10%	30%	50%	20%	40%	60%	80%	10%	30%	50%	20%	40%	60%	80%
TS [10]	TPR	0.28	0.14	0.12	0.23	0.14	0.15	0.15	0.40	0.36	0.27	0.38	0.32	0.30	0.29
	FPR	0.04	0.07	0.09	0.08	0.06	0.07	0.09	**0.04**	**0.06**	**0.06**	**0.03**	**0.02**	0.02	**0.06**
NL [23]	TPR	0.80	0.75	0.64	0.83	0.77	0.34	-	0.82	0.81	0.75	0.80	0.79	0.66	-
	FPR	0.09	0.26	0.34	0.03	**0.03**	**0.00**	-	0.10	0.18	0.22	**0.03**	0.03	**0.01**	-
Ours	TPR	**0.82**	**0.90**	**0.87**	**0.89**	**0.94**	**0.92**	**0.83**	**0.83**	**0.86**	**0.85**	**0.89**	**0.90**	**0.87**	**0.79**
	FPR	**0.02**	**0.04**	**0.11**	**0.02**	0.05	0.05	0.05	0.07	0.23	0.32	0.04	0.06	0.06	**0.06**

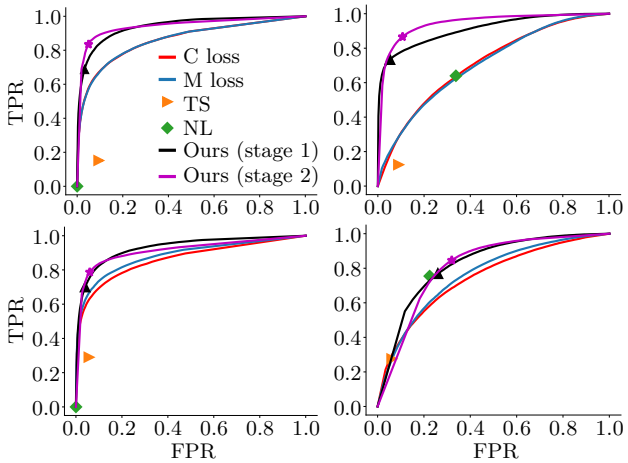

Figure 3. Label noise detection methods in ImageNet32. Left: 80% uniform ID (top) and OOD (bottom) noise. Right: 50% non-uniform ID (top) and OOD (bottom) noise. The proposed method (star in stage 2 denotes operating point) surpasses detection based on cross-entropy loss without (C) and with mixup (M), TS [10] and NL [23]. Key: ID (in-distribution). OOD (out-of-distribution).

curves for our label noise detection method and the small loss trick (using cross-entropy with and without mixup [46]). The proposed approach clearly outperforms straightforward application of the small loss trick as is usually done by many recent works [14, 2, 34]. It also outperforms two recently proposed label noise detection methods [10, 23], showing consistent improvements across different label noise distributions (see Table 2). Note that we encounter some limitations addressing high levels of non-uniform out-of-distribution noise, which occurs due to the nature of the classes used as noise. We are using the ImageNet confusion matrix in the validation set to introduce noise and we have 100 (900) in-(out-of-) distribution classes, thus using the most challenging classes as out-of-distribution noise.

5.3. Comparison with related work

We select representative top-performing loss correction [30, 2], relabeling [36], and label noise robust regularization approaches [46] to compare against our label noise Distribution Robust Pseudo-Labeling (DRPL) approach in Table 3. The proposed method gives remarkable improvements across different levels and distributions of label noise. Note that, unlike most methods compared, our method shows little degradation between the best (reported in Table 3) and last epoch accuracy (see extended results in the supplementary material). In general, R [36] behaves consistently across label noise levels and distributions, while DB [2] has problems with non-uniform noise. FW [30] and M [46] tend to exhibit worse performance. An important observation is that non-uniform out-of-distribution noise exhibits much less degradation than other noise types for all methods. This is reasonable as out-of-distribution samples whose content is close to an in-distribution class will contribute to improved representation learning due to semantic similarities, and the network predictions for these samples will not harm the model for in-distribution classes. This behavior resembles real-world noisy data as observed in the WebVision dataset results in Subsection 5.5.

Table 3. ImageNet32 (64) accuracy on top (bottom). Bold denotes best performance. Key: NU (non-uniform); U (uniform); ID (in-distribution); OOD (out-of-distribution).

	NU-ID		U-ID		NU-OOD		U-OOD	
	30%	50%	40%	80%	30%	50%	40%	80%
FW [30]	54.22	43.38	52.06	31.20	62.14	55.06	56.32	40.08
R [36]	67.24	63.62	62.98	41.52	66.36	62.80	64.04	45.00
M [46]	67.14	51.96	61.98	38.92	66.14	60.62	64.66	47.40
DB [2]	62.88	52.20	67.62	45.34	64.86	60.58	65.96	39.30
DRPL (ours)	**73.46**	**68.18**	**73.48**	**61.78**	**71.38**	**67.32**	**71.36**	**54.10**
	NU-ID		U-ID		NU-OOD		U-OOD	
	30%	50%	40%	80%	30%	50%	40%	80%
FW [30]	60.10	46.06	57.42	37.84	69.86	63.38	63.08	47.68
R [36]	74.28	69.20	70.98	48.44	74.22	70.74	72.78	54.00
M [46]	74.02	58.14	69.90	49.22	74.78	69.40	73.94	59.54
DB [2]	71.30	60.98	74.56	56.44	77.94	70.38	74.08	50.98
DRPL (ours)	**81.90**	**77.66**	**81.50**	**73.08**	**80.44**	**76.38**	**79.76**	**64.34**

5.4. Effects of label noise memorization

We know that clean (easier) patterns are learned first by deep neural networks and that noisy labels can be completely memorized [3, 45]. *How do networks memorize noisy sam-*

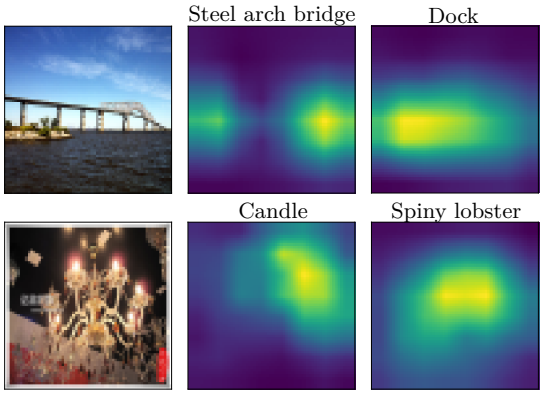

Figure 4. Visualization of label noise memorization for undetected noisy images in ImageNet64 (50%, non-uniform in-distribution noise). From left to right: image and Class Activation Map (CAM) for the true class and the predicted class. Note that the predicted class (right) is also the noisy label used during training.

ples? Figure 4 provides some intuition. It shows the Class Activation Maps [50] for the true class and the predicted class for undetected noisy samples. The network skips relevant areas for the true class, attending to areas that might help explain (i.e. memorizing) the noisy label. For example, the network focuses on the left part of the bridge instead of right to better fit the noisy label *dock* (first row), and the whole lamp instead of the candles for the the noisy label *lobster* (second row). Further examples are provided in the supplementary material.

Does memorization negatively impact visual representation learning? We follow the standard approach of using linear probes [47] to verify the utility of features under different noise levels and distributions in a target task. Specifically, we train a linear classifier on the global average pooled activations obtained after each of the 4 PreAct ResNet-18 blocks. Figure 5 presents the results in ImageNet64. Our model (O) clearly outperforms mixup (M) in the target task performance using features from the last block. Better source models (i.e. trained with less noise) also tend to produce better target performance. However, an interesting finding is that for both uniform and non-uniform noise, the final accuracy exhibits degradation in L4, while for earlier features no degradation is observed even for M (a model that has memorized the noise). An exception is 80% of uniform noise, where degradation is found across all blocks, but with more discriminative representations learned by our approach. Similar results are observed in ImageNet32 (see supplementary material).

5.5. Other datasets and real-world frameworks

Table 4 compares our DRPL approach with related work in CIFAR-10/100, showing improved performance over all other approaches (extended results in the supplementary material). We evaluate the same approaches as in ImageNet32/64 [30, 36, 46, 2] and also add further recent approaches [43, 44, 42]. The figures given are from our own evaluation runs (details in the supplementary material). Improved results for uniform and non-uniform noise in CIFAR confirm the improvements observed in ImageNet32/64. The additional recent approaches, GCE [48], DMI [42], and PEN [44] are outperformed by our approach DRPL in most cases (with the exception being PEN for low noise levels in CIFAR-100) with GCE and DMI far from top-performance.

We also compare DRPL against related work in the first 50 classes of WebVision 1.0 dataset [27] to verify the practical use of our method in real-world noise scenarios. Table 4 shows the proposed approach is more accurate than all compared approaches. Note, however, that many approaches give very similar performance. Surprisingly, a straightforward approach like M [46], not specifically designed to deal with label noise, gives the second highest accuracy. The similar performance among approaches is something that can be also observed for non-uniform out-of-distribution noise in ImageNet32/64. Another similarity here with non-uniform out-of-distribution noise is the small amount of degradation in performance at the last training epoch (reported in the supplementary material for all comparisons). Furthermore, the reduced noise level in WebVision, estimated at around 20% in [27], and with 17% of detected noisy samples in our 50 class subset, helps to explain performance similarities across approaches: this is also seen in CIFAR and ImageNet32/64 with low noise levels. We present detected noisy images in Figure 9 (more examples are provided in the supplementary material), which are mainly from outside the distribution.

6. Discussion

The proposed label noise detection method outperforms other state-of-the-art across label noise distributions (see Subsection 5.2) and its combination with a modern semi-supervised detection technique [1] surpasses many recent approaches (see Tables 3 and 4). When the proposed pipeline is applied in a scenario with little or no noise, however, performance suffers an important drop. For example, Table 4 shows top performances of 78.33 for [46] and 76.31 for [44] in CIFAR-100 with 0% and 10% noise, while our approach achieves 72.27 and 72.40. These approaches do not discard samples, suggesting that the information discarded by our approach is important for achieving top performance. This highlights that high loss is indicative not only of incorrect labels, but also of difficult samples [21]. Important information can be discarded when high loss is used to discard labels of presumed noisy samples, the semi-supervised approach determines whether this information is recovered.

In addition to discarded labels, undetected noisy samples also require consideration. Including a noisy sample in the labeled set corrupts it and harms learning from unlabeled data. Subsection 5.4 investigated network behavior with respect to these samples, showing how the network memo-

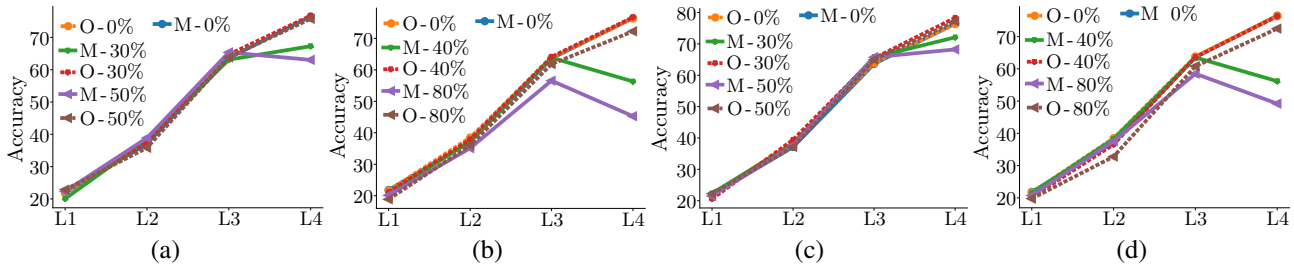

Figure 5. ImageNet64 linear probes. A linear classifier is trained using features at different depths. Source task: 100 classes. Target task: 25 classes from the remaining 900. Noise: in-distribution non-uniform (a) and uniform (b), and out-of-distribution non-uniform (c) and uniform (d). Model from the last training epoch in the source domain is used (i.e. M has fitted the noise). Key: M: Mixup. O: Ours.

Table 4. CIFAR-10/100 and WebVision (50 classes) accuracy. Key: NU (non-uniform noise). U (uniform noise). CE: Cross-entropy.

	CIFAR-10								CIFAR-100								WebV
	0%	NU			U				0%	NU			U				
		10%	30%	40%	20%	40%	60%	80%		10%	30%	40%	20%	40%	60%	80%	
CE	93.87	90.97	90.22	88.16	87.75	83.32	75.71	43.69	74.59	68.18	54.20	46.55	59.19	51.44	39.05	19.59	73.88
FW [30]	94.61	90.85	87.95	84.85	85.46	80.67	70.86	45.58	75.43	68.82	54.65	45.38	61.03	51.44	39.05	19.59	74.68
R [36]	94.37	93.70	92.69	92.70	92.77	89.97	85.62	53.26	74.43	73.09	68.25	59.49	70.79	66.35	57.48	30.65	76.52
M [46]	**96.07**	93.79	91.38	87.01	91.27	85.84	80.78	57.93	**78.33**	73.39	59.15	49.40	66.60	54.69	45.80	27.02	80.76
GCE [48]	93.93	91.40	90.45	88.39	88.49	84.09	76.55	43.39	74.91	68.34	55.56	47.24	60.09	61.23	49.75	25.77	74.28
DB [2]	92.78	91.77	93.23	91.25	93.95	92.38	**89.53**	49.90	70.64	68.19	62.81	55.76	69.12	64.84	57.85	46.45	79.68
DMI [42]	93.93	91.31	91.34	88.64	88.40	83.98	75.91	44.17	74.75	68.29	54.40	46.65	59.16	53.49	41.49	20.50	73.96
PEN [44]	93.94	93.19	92.94	91.63	92.87	91.34	89.15	56.14	77.80	**76.31**	63.67	50.64	**75.16**	69.56	56.16	27.12	79.96
DRPL (ours)	94.47	**95.70**	**93.65**	**93.14**	**94.20**	**92.92**	89.21	**64.35**	72.27	72.40	**69.30**	**65.86**	71.25	**73.13**	**68.71**	**53.04**	**82.08**

Figure 6. Examples of detected noisy images in WebVision [27]. The first column shows a clean example from the class.

rizes the noisy labels (Figure 4) and that this memorization is detrimental to the formation of discriminative features in intermediate layers (Figure 5). These results open the door to tackling noise in different ways at different depths.

Methods behave differently against artificially introduced noise in CIFAR than in real-world scenarios like WebVision [27]. This is clear from the performance in the last epoch for all methods (reported the extension of Table 4 in the supplementary material), where many of them are overfit to label noise. Substantial accuracy degradation is seen in many methods in CIFAR, while in WebVision degradations are minor and mixup (M) [46], which does not deal specifically with label noise, outperforms most approaches. We believe that evaluation with out-of-distribution noise in ImageNet32/64 provides a better understanding of real-world noise. Moreover, simplifying label noise to a particular distribution might be unrealistic and a combination of uniform and non-uniform in and out-of-distribution noise might be better (e.g. the first row in Figure 9 presents a shark (in-distribution class) and other images (out-of-distribution) as noisy samples). It would be worth studying conditioning noise on specific class subsets, rather than overall class-conditional flips, as classes have different prototype samples [15] that may be more or less difficult to fit for other classes.

7. Conclusion

We propose a framework to study multiple label noise distributions and a straightforward approach based on label noise detection and semi-supervised learning to tackle them all. We show that, in ideal conditions, semi-supervised learning outperforms loss correction based on an oracle label noise transition matrix. We also provide intuitions about the network behavior when memorizing noisy labels and show that such memorization does not often harm learning discriminative intermediate representations. Results in five datasets support the generality and robustness of our approach and help in better understanding real-world label noise.

Acknowledgements

This work was supported by Science Foundation Ireland (grant numbers SFI/15/SIRG/3283 and SFI/12/RC/2289).

References

[1] E. Arazo, D. Ortego, P. Albert, N.E. O'Connor, and K. McGuinness. Pseudo-Labeling and Confirmation Bias in Deep Semi-Supervised Learning. *arXiv: 1908.02983*, 2019. 1, 3, 3.1, 3.3, 4, 6, 8.2

[2] E. Arazo, D. Ortego, P. Albert, N. O'Connor, and K. McGuinness. Unsupervised Label Noise Modeling and Loss Correction. In *International Conference on Machine Learning (ICML)*, 2019. 1, 2, 3.1, 3.1, 5.2, 5.3, 3, 5.5, 4, 8.2, 5, 6, 7, 8.6

[3] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M.S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, and S. Lacoste-Julien. A Closer Look at Memorization in Deep Networks. In *International Conference on Machine Learning (ICML)*, 2017. 1, 3.1, 5.4

[4] W.H. Beluch, T. Genewein, A. Nürnberger, and J.M. Köhler. The Power of Ensembles for Active Learning in Image Classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

[5] Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013. 1

[6] D. Berthelot, N. Carlini, I.J. Goodfellow, N. Papernot, A. Oliver, and C. Raffel. MixMatch: A Holistic Approach to Semi-Supervised Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 1

[7] P. Chen, B. Liao, G. Chen, and S. Zhang. Understanding and Utilizing Deep Neural Networks Trained with Noisy Labels. In *International Conference on Machine Learning (ICML)*, 2019. 1, 4.3

[8] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 4.1

[9] D. DeTone, T. Malisiewicz, and A. Rabinovich. Deep image homography estimation. *arXiv:1606.03798*, 2016. 1

[10] Y. Ding, L. Wang, D. Fan, and B. Gong. A Semi-Supervised Two-Stage Approach to Learning from Noisy Labels. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018. 1, 2, 5.2, 3

[11] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional Two-stream Network Fusion for Video Action Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[12] B. Frenay and M. Verleysen. Classification in the Presence of Label Noise: A Survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5):845–869, 2014. 2

[13] S. Guo, W. Huang, H. Zhang, C. Zhuang, D. Dong, M.R. Scott, and D. Huang. CurriculumNet: Weakly Supervised Learning from Large-Scale Web Images. In *European Conference on Computer Vision (ECCV)*, 2018. 2

[14] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 1, 3.1, 5.2

[15] J. Han, P. Luo, and X. Wang. Deep Self-Learning From Noisy Labels. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 2, 6

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4, 4.1

[17] K. He, X. Zhang, S. Ren, and J. Sun. Identity Mappings in Deep Residual Networks. In *European Conference on Computer Vision (ECCV)*, 2016. 1, 4

[18] D. Hendrycks, K. Lee, and M. Mazeika. Using Pre-Training Can Improve Model Robustness and Uncertainty. In *International Conference on Machine Learning (ICML)*, 2019. 1

[19] Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. Using Trusted Data to Train Deep Networks on Labels Corrupted by Severe Noise. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 5.1

[20] G. Huang, Z. Liu, L. Van der Maaten, and K.Q. Weinberger. Densely Connected Convolutional Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4.1

[21] A.H. Jiang, D.L.-K. Wong, G. Zhou, D.G. Andersen, J. Dean, G.R. Ganger, G. Joshi, M. Kaminksy, M. Kozuch, Z.C. Lipton, and P. Pillai. Accelerating Deep Learning by Focusing on the Biggest Losers. *arXiv: 1910.00762*, 2019. 6

[22] L. Jiang, Z. Zhou, T. Leung, L.J. Li, and L. Fei-Fei. MentorNet: Learning Data-Driven Curriculum for Very Deep Neural Networks on Corrupted Labels. In *International Conference on Machine Learning (ICML)*, 2018. 1, 2, 4.3

[23] Y. Kim, J. Yim, J. Yun, and J. Kim. NLNL: Negative Learning for Noisy Labels. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 1, 2, 5.2, 3

[24] A. Kolesnikov, X. Zhai, and L. Beyer. Revisiting Self-Supervised Visual Representation Learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[25] R. Krishna, K. Hata, F. Ren, L. Fei-Fei, and J. C. Niebles. Dense-Captioning Events in Videos. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 1

[26] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 1, 4, 4.2

[27] W. Li, L. Wang, W. Li, E. Agustsson, and L. Van Gool. WebVision Database: Visual Learning and Understanding from Web Data. *arXiv: 1708.02862*, 2017. 1, 2, 4, 4.3, 5.5, 6, 6

[28] Y. Ono, E. Trulls, P. Fua, and K. Moo Yi. LF-Net: Learning Local Features from Images. *arXiv: 1805.09662*, 2018. 1

[29] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan. Learning Features by Watching Objects Move. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[30] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu. Making Deep Neural Networks Robust to Label Noise: A Loss Correction Approach. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 4.1, 5.1, 5.1, 5.3, 3, 5.5, 4, 8.2, 5, 6, 7, 8.6

[31] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[32] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *International Conference on Learning Representations (ICLR)*, 2015. 1, 2

[33] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv: 1409.1556*, 2014. 4.1

[34] H. Song, M. Kim, and J.-G. Lee. SELFIE: Refurbishing Unclean Samples for Robust Deep Learning. In *International Conference on Machine Learning (ICML)*, 2019. 1, 3.1, 5.2

[35] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the Inception Architecture for Computer Vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4.1

[36] 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. 1, 2, 3.1, 5.3, 3, 5.5, 4, 8.2, 5, 6, 7, 8.6

[37] Arash Vahdat. Toward Robustness against Label Noise in Training Deep Discriminative Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 2

[38] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie. Learning From Noisy Large-Scale Datasets With Minimal Supervision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[39] M. Wang and W. Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018. 1

[40] Y. Wang, W. Liu, X. Ma, J. Bailey, H. Zha, L. Song, and S.-T. Xia. Iterative Learning With Open-Set Noisy Labels. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 4.1

[41] X. Xia, T. Liu, N. Wang, B. Han, C. Gong, G. Niu, and M. Sugiyama. Are Anchor Points Really Indispensable in Label-Noise Learning? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 5.1

[42] Y. Xu, P. Cao, Y. Kong, and Y. Wang. L_DMI: An Information-theoretic Noise-robust Loss Function. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 1, 2, 5.5, 4, 8.2, 7, 8.6

[43] J. Yao, H. Wu, Y. Zhang, I.W Tsang, and J. Sun. Safeguarded Dynamic Label Regression for Noisy Supervision. In *Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence (AAAI)*, 2019. 1, 2, 5.1, 5.5

[44] K. Yi and J. Wu. Probabilistic End-To-End Noise Correction for Learning With Noisy Labels. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2, 4.1, 4.2, 5.5, 6, 4, 8.2, 7, 8.6

[45] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires re-thinking generalization. In *International Conference on Learning Representations (ICLR)*, 2017. 1, 5.4

[46] H. Zhang, M. Cisse, Y.N. Dauphin, and D. Lopez-Paz. mixup: Beyond Empirical Risk Minimization. In *International Conference on Learning Representations (ICLR)*, 2018. 3.3, 5.2, 5.3, 3, 5.5, 6, 4, 8.2, 5, 6, 7, 8.6

[47] R. Zhang, P. Isola, and A. Efros. Split-Brain Autoencoders: Unsupervised Learning by Cross-Channel Prediction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5.4

[48] Z. Zhang and M. Sabuncu. Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 2, 5.5, 4, 7, 8.6

[49] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid Scene Parsing Network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[50] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning Deep Features for Discriminative Localization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5.4, 8.3

8. Supplementary material for: Towards Robust Learning with Different Label Noise Distributions

8.1. Proposed method: algorithm

Algorithm 1 summarizes the proposed label noise Distribution Robust Pseudo-Labeling (DRPL) approach to deal with label noise.

Algorithm 1 Summary of proposed method.

Input: Dataset $\mathcal{D}$ with potentially noisy labels y.
Output: Model $h_\theta(x)$, clean $\mathcal{D}_l$ and noisy $\mathcal{D}_u$ samples.

 # Label noise detection: first stage
Relabeling: train model $h_\phi(x)$ on $\mathcal{D}$ using Eq. 1
Compute cross-entropy loss $\ell^*(\phi)$ w.r.t. original labels y.
Initial noise model: fit two-component BMM to $\ell^*(\phi)$.
Initial labeled/unlabeled sets: create $\mathcal{D}'_l$ and $\mathcal{D}'_u$ using the initial BMM (Eqs. 2 and 3).
 # Label noise detection: second stage
Train $h_\varphi(x)$ in a semi-supervised manner using $\mathcal{D}'_l$ and $\mathcal{D}'_u$.
Compute cross-entropy loss $\ell^*(\varphi)$ w.r.t. original labels y.
Final noise model: fit two-component BMM to $\ell^*(\varphi)$.
Final labeled/unlabeled sets: create $\mathcal{D}_l$ and $\mathcal{D}_u$ using the final BMM as in Eqs. 2 and 3.
 # Semi-supervised learning
Train $h_\theta(x)$ in a semi-supervised manner using $\mathcal{D}_l$ and $\mathcal{D}_u$.

8.2. Extended implementation details

The λ_1 and λ_2 hyperparameters are set as in [36]; we have not sought careful tuning. We use the default parameters for the semi-supervised method [1]. Two important hyperparameters in our method are the thresholds γ_1 and γ_2 to detect label noise in each label noise detection stage. In the first stage, the idea is to get sufficient data for semi-supervised learning with labels that are as clean as possible. We select $\gamma_1 = 0.05$ in ImageNet32/64 and WebVision, and $\gamma_1 = 0.1$ in CIFAR. We slightly increase the threshold in CIFAR to assure that enough data is selected to perform a successful semi-supervised learning, as we had some problems in CIFAR-100 to select enough data (less than 4K samples were selected occasionally in CIFAR-100, which according to results in [1] is not enough to prevent performance degradation). We keep the same configuration in CIFAR-10 to demonstrate its generality.

Training details for compared methods:

- F [30]: In ImageNet32/64 and CIFAR we train 200 epochs with initial learning rate of 0.1 that we divide by 10 in epochs 100 and 150. For WebVision we train 125 epochs and reduce the learning rate in epochs 75 and 120. Forward correction always starts in epoch 50.
- M [46]: In ImageNet32/64 and CIFAR we train 300 epochs with initial learning rate of 0.1 that we divide by 10 in epochs 100 and 250. For WebVision we train 200 epochs and reduce the learning rate in epochs 75 and 120. Mixup parameter α is set to 1.
- R [36]: In ImageNet32/64 and CIFAR we train in the first stage of the method as in the first stage of our label noise detection method, i.e. we train for 100 epochs with initial learning rate of 0.1 that we divide by 10 in epochs 45 and 80. Relabeling starts in epoch 40. For the second stage we train 120 epochs with initial learning rate of 0.1 that we divide by 10 in epochs 40 and 80. Other hyperparameters are kept as in [36].
- DB [2]: We use the code[1] associated to the official implementation of [2]. We keep the default configuration used for CIFAR-10/100 and use it also in ImageNet32/64. For WebVision we train for 200 epochs with initial learning rate of 0.1 that we divide by 10 in epochs 100 and 175 (bootstrapping starts in epoch 102).
- GCE and DMI [42]: We use the code[2] associated to the official implementation of [42]. We keep the default configuration used for CIFAR-10 for both CIFAR-10 and CIFAR-100. We respect the use of the best model in the pre-training phase using cross-entropy loss, although such selection would not be straightforward without using a clean set in a real scenario.
- P [44]: We use the official implementation[3] of [44]. We found difficulties on configuring this method for the different datasets, as similar configurations did not work in CIFAR-10/100, and WebVision. We always use $\alpha = 0.1$, $\beta = 0.4$, and learning rate of 0.1 in the first and second stages, whereas we use 0.2 as starting learning rate in the third stage. For CIFAR-10 different hyperparameters are used in [44] to deal with different noise distributions and noise levels, which we did not find reasonable. Therefore, we set a common configuration with $\lambda = 600$ and training the suggested number of epochs [44]. In CIFAR-100 we tried the configuration suggested in the paper and it did not converge to reasonable performance, thus we reduced the learning rate from 0.35 to 0.1 to make it stable and use $\lambda = 10000$ as suggested. For WebVision we used the same configuration used in CIFAR-10 and trained 40 epochs in the first stage, 60 in the second and 100 in the third. Note that we tried the suggested CIFAR-100 configuration in WebVision, but it led to poor performance.

[1] https://git.io/fjsvE
[2] https://git.io/JeRGh
[3] https://git.io/JezIO

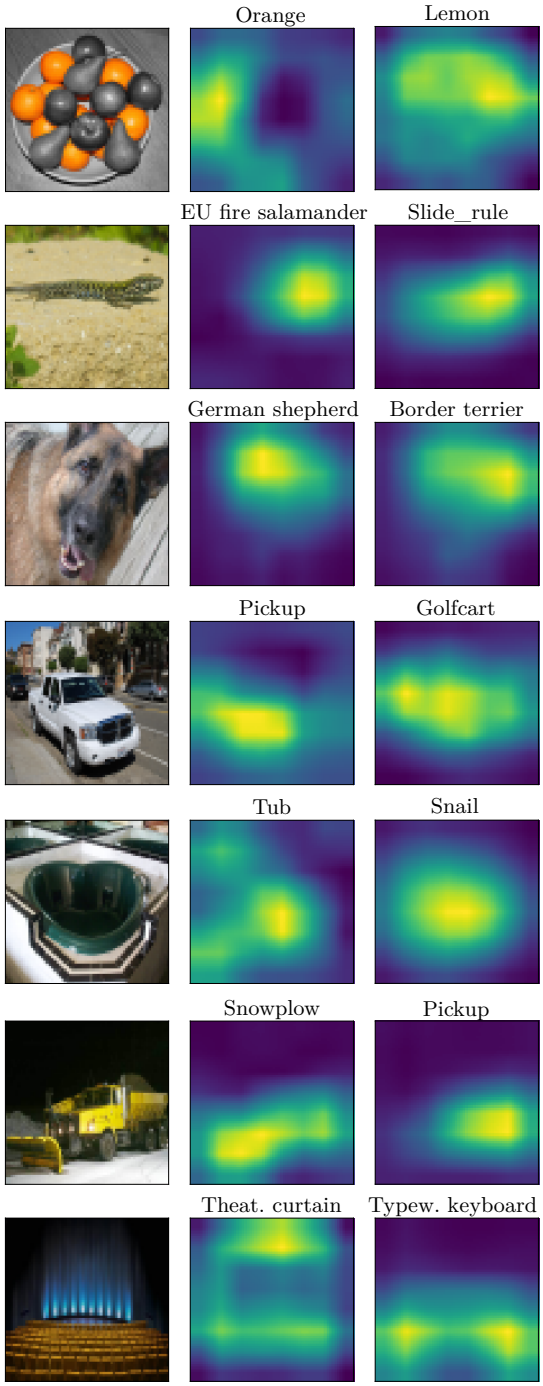

Figure 7. Visualization of label noise memorization for undetected noisy images in ImageNet64 (50%, non-uniform in-distribution noise). From left to right: image and Class Activation Map (CAM) for the true class and the predicted class. Note that the predicted class (right) is also the noisy label used during training.

8.3. Extended results: Label noise memorization

Figure 7 presents Class Activation Maps [50] for the true class (middle) and the predicted class (right) for undetected noisy samples. As shown in Figure 4 of the paper, the network skips relevant areas for the true class, and shifts attention to areas that might help in explaining (i.e. memorizing) the noisy label. The network extends the class activation maps to cover image regions that share visual similarities with the incorrect class, while other times it omits characteristic areas of the true class. The former situation can be observed in rows one to four where important areas are expanded, whereas the latter is seen in the last two rows. The blades of the snowplow are possibly skipped to help with explaining the noisy label *pickup* and the theater curtain is skipped and attention is focused on the theater's seats which resemble the noisy label *keyboard*.

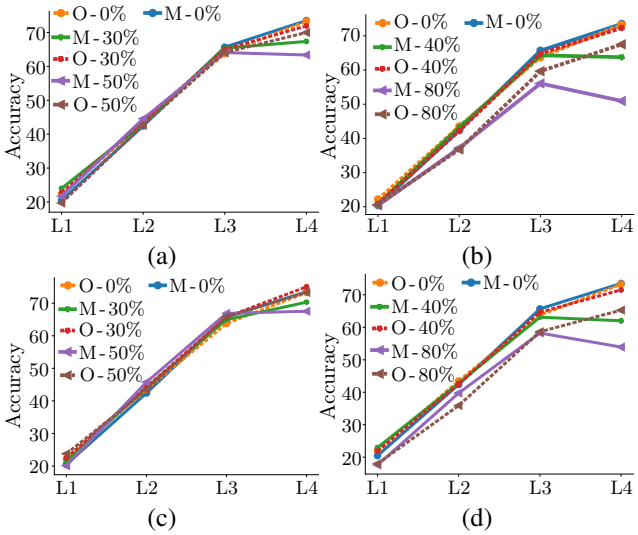

Figure 8. ImageNet32 linear probes. A linear classifier is trained using features at different depths. Source domain: 100 classes. Target domain: 25 classes from the remaining 900. Noise: in-distribution non-uniform (a) and uniform (b), and out-of-distribution non-uniform (c) and uniform (d). Model from the last training epoch in the source domain is used (i.e. M has fitted the noise). Key: M: Mixup. O: Ours.

Figure 8 reports the results of the linear probes experiment in ImageNet32, showing similar characteristics as those in ImageNet64 reported in the paper. Similar conclusions can be extracted, highlighting little or no degradation of intermediate features when learning with label noise.

8.4. Extended results: ImageNet32/64

Tables 5 and 6 report the extended results for ImageNet32/64 with four different label noise distributions, more noise levels, and include accuracy in the last training epoch (useful to understand when methods fit label noise).

8.5. Extended results: CIFAR-10/100

Table 8.5 gives additional results for CIFAR-10/100 adding the performance in the last epoch, which as found in

ImageNet32/64, reveals methods degradation due to label noise memorization (e.g. CE, M).

8.6. Extended results: WebVision

Table 8 extends the results obtained in WebVision (first 50 classes) by including the performance in the last training epoch when training with a ResNet-18 from scratch. Figure 9 shows examples of detected noisy images.

Table 5. ImageNet32 accuracy for 100 classes. Bold denotes best performance. Key: NU (Non-uniform). U (Uniform). ID (in-distribution). OOD (out-of-distribution).

		0%	NU-ID			U-ID				NU-OOD			U-OOD			
			10%	30%	50%	20%	40%	60%	80%	10%	30%	50%	20%	40%	60%	80%
FW [30]	*Best*	71.84	65.84	54.22	43.38	57.40	52.06	44.36	31.20	67.50	62.14	55.06	63.04	56.32	50.16	40.08
	Last	71.42	65.84	52.00	33.88	57.30	42.48	25.28	11.02	67.20	61.90	54.00	62.80	53.58	42.86	26.42
R [36]	*Best*	69.26	68.64	67.24	63.62	66.66	62.98	57.54	41.52	68.28	66.36	62.80	67.20	64.04	58.92	45.00
	Last	69.06	68.40	66.98	63.22	66.30	62.56	57.34	41.24	67.96	66.28	62.48	67.20	63.86	58.54	44.64
M [46]	*Best*	**74.80**	71.64	67.14	51.96	67.76	61.98	53.84	38.92	71.10	66.14	60.62	69.88	64.66	58.48	47.40
	Last	**74.80**	69.74	57.00	37.56	62.72	48.40	31.30	12.32	70.76	64.68	56.38	65.50	56.80	44.84	26.86
DB [2]	*Best*	66.18	67.54	62.88	52.20	68.42	67.62	63.56	45.34	67.40	64.86	60.58	67.28	65.96	59.26	39.30
	Last	65.42	66.48	61.88	51.64	67.84	62.98	62.72	44.56	66.90	64.08	60.18	66.72	65.28	58.86	38.44
DRPL (ours)	*Best*	72.98	**72.60**	**73.46**	**68.18**	**74.28**	**73.48**	**70.06**	**61.78**	**72.60**	**71.38**	**67.32**	**73.72**	**71.36**	**65.52**	**54.10**
	Last	72.60	**72.34**	**73.00**	**67.56**	**73.80**	**73.30**	**70.06**	**61.30**	**72.26**	**71.36**	**67.32**	**73.20**	**70.64**	**65.18**	**53.16**

Table 6. ImageNet64 accuracy for 100 classes. Bold denotes best performance. Key: NU (Non-uniform). U (Uniform). ID (in-distribution). OOD (out-of-distribution).

		0%	NU-ID			U-ID				NU-OOD			U-OOD			
			10%	30%	50%	20%	40%	60%	80%	10%	30%	50%	20%	40%	60%	80%
FW [30]	*Best*	79.80	73.08	60.10	46.06	64.06	57.42	51.52	37.84	73.08	69.86	63.38	70.86	63.08	58.14	47.68
	Last	79.32	73.08	60.10	42.96	63.92	49.70	31.28	14.18	73.08	69.42	62.50	70.60	61.78	51.74	35.56
R [36]	*Best*	76.46	75.94	74.28	69.20	73.70	70.98	65.08	48.44	75.90	74.22	70.74	75.54	72.78	66.90	54.00
	Last	76.26	75.28	74.02	68.84	73.40	70.74	65.04	48.28	75.48	73.70	70.56	75.22	72.48	66.64	53.72
M [46]	*Best*	**83.38**	80.08	74.02	58.14	76.34	69.90	60.96	49.22	79.88	74.78	69.40	78.44	73.94	67.86	59.54
	Last	**82.88**	78.42	63.54	44.56	70.86	54.60	34.12	14.12	79.14	72.92	64.76	73.26	64.00	51.16	33.28
DB [2]	*Best*	76.16	75.38	71.30	60.98	75.86	74.56	71.50	56.44	80.92	77.94	70.38	75.24	74.08	68.32	50.98
	Last	71.48	72.70	70.66	58.70	74.38	73.76	70.00	55.74	80.76	77.96	70.24	72.54	73.52	67.50	50.78
DRPL (ours)	*Best*	81.22	**81.84**	**81.90**	**77.66**	**82.74**	**81.50**	**79.66**	**73.08**	**82.12**	**80.44**	**76.38**	**82.78**	**79.76**	**75.72**	**64.24**
	Last	80.78	**81.54**	**81.40**	**77.08**	**82.20**	**81.20**	**79.28**	**72.60**	**81.80**	**80.34**	**75.90**	**82.40**	**79.46**	**75.18**	**63.28**

Table 7. CIFAR-10/100 accuracy. Key: NU (non-uniform noise). U (uniform noise). CE: Cross-entropy. Best (last) denotes the accuracy obtained in the best (last) epoch.

		CIFAR-10								CIFAR-100							
		0%	NU			U				0%	NU			U			
			10%	30%	40%	20%	40%	60%	80%		10%	30%	40%	20%	40%	60%	80%
CE	*Best*	93.87	90.97	90.22	88.16	87.75	83.32	75.71	43.69	74.59	68.18	54.20	46.55	59.19	51.44	39.05	19.59
	Last	93.85	88.81	81.69	76.04	78.93	55.06	36.75	33.09	74.34	68.10	53.28	44.46	58.75	42.92	25.33	8.29
FW [30]	*Best*	94.61	90.85	87.95	84.85	85.46	80.67	70.86	45.58	75.43	68.82	54.65	45.38	61.03	51.44	39.05	19.59
	Last	94.45	90.39	81.87	76.65	81.03	61.06	42.16	24.09	75.37	68.66	54.42	45.32	60.83	45.31	26.79	8.44
R [36]	*Best*	94.37	93.70	92.69	92.70	92.77	89.97	85.62	53.26	74.43	73.09	68.25	59.49	70.79	66.35	57.48	30.65
	Last	94.21	93.61	92.52	92.54	92.58	89.96	85.42	52.96	74.20	72.71	68.13	59.41	70.70	66.18	57.22	30.53
M [46]	*Best*	**96.07**	93.79	91.38	87.01	91.27	85.84	80.78	57.93	**78.33**	73.39	59.15	49.40	66.60	54.69	45.80	27.02
	Last	95.96	93.30	83.26	77.74	84.76	66.07	43.95	20.38	77.90	72.40	57.63	48.07	66.40	52.20	33.73	13.21
GCE [48]	*Best*	93.93	91.40	90.45	88.39	88.49	84.09	76.55	43.39	74.91	68.34	55.56	47.24	60.09	61.23	49.75	25.77
	Last	93.85	89.48	80.60	76.04	82.95	80.84	66.95	20.19	74.68	68.01	52.84	45.02	59.49	55.50	39.96	21.65
DB [2]	*Best*	92.78	91.77	93.23	91.25	93.95	92.38	**89.53**	49.90	70.64	68.19	62.81	55.76	69.12	64.84	57.85	46.45
	Last	79.18	89.58	92.20	91.20	93.82	92.26	**89.15**	15.53	64.79	67.09	58.59	47.44	69.11	62.78	52.33	45.67
DMI [42]	*Best*	93.93	91.31	91.34	88.64	88.40	83.98	75.91	44.17	74.75	68.29	54.40	46.65	59.16	53.49	41.49	20.50
	Last	93.88	91.11	91.16	83.99	88.33	83.24	14.78	43.67	74.44	68.15	54.15	46.20	58.82	53.22	41.29	20.30
PEN [44]	*Best*	93.94	93.19	92.94	91.63	92.87	91.34	89.15	56.14	77.80	**76.31**	63.67	50.64	**75.16**	69.56	56.16	27.12
	Last	93.89	93.14	92.85	91.57	92.72	91.32	89.05	55.99	77.75	76.05	59.29	48.26	74.93	68.49	53.08	25.41
DRPL (ours)	*Best*	94.47	**95.70**	**93.65**	**93.14**	**94.20**	**92.92**	89.21	**64.35**	72.27	72.40	**69.30**	**65.86**	71.25	**73.13**	**68.71**	**53.04**
	Last	94.08	**95.50**	**92.98**	**92.84**	**94.00**	**92.27**	87.23	**61.07**	71.84	72.03	**69.30**	**65.69**	71.16	**72.37**	**68.42**	**52.95**

Table 8. Top-1 accuracy in first 50 classes of WebVision. We train all methods from scratch.

	CE	FW [30]	R [36]	M [46]	GCE [48]	DB [2]	DMI [42]	P[44]	DRPL (ours)
Best	73.88	74.68	76.52	80.76	74.28	79.68	73.96	79.96	**82.08**
Last	73.76	74.32	76.24	79.96	74.08	79.56	73.88	79.44	**82.00**

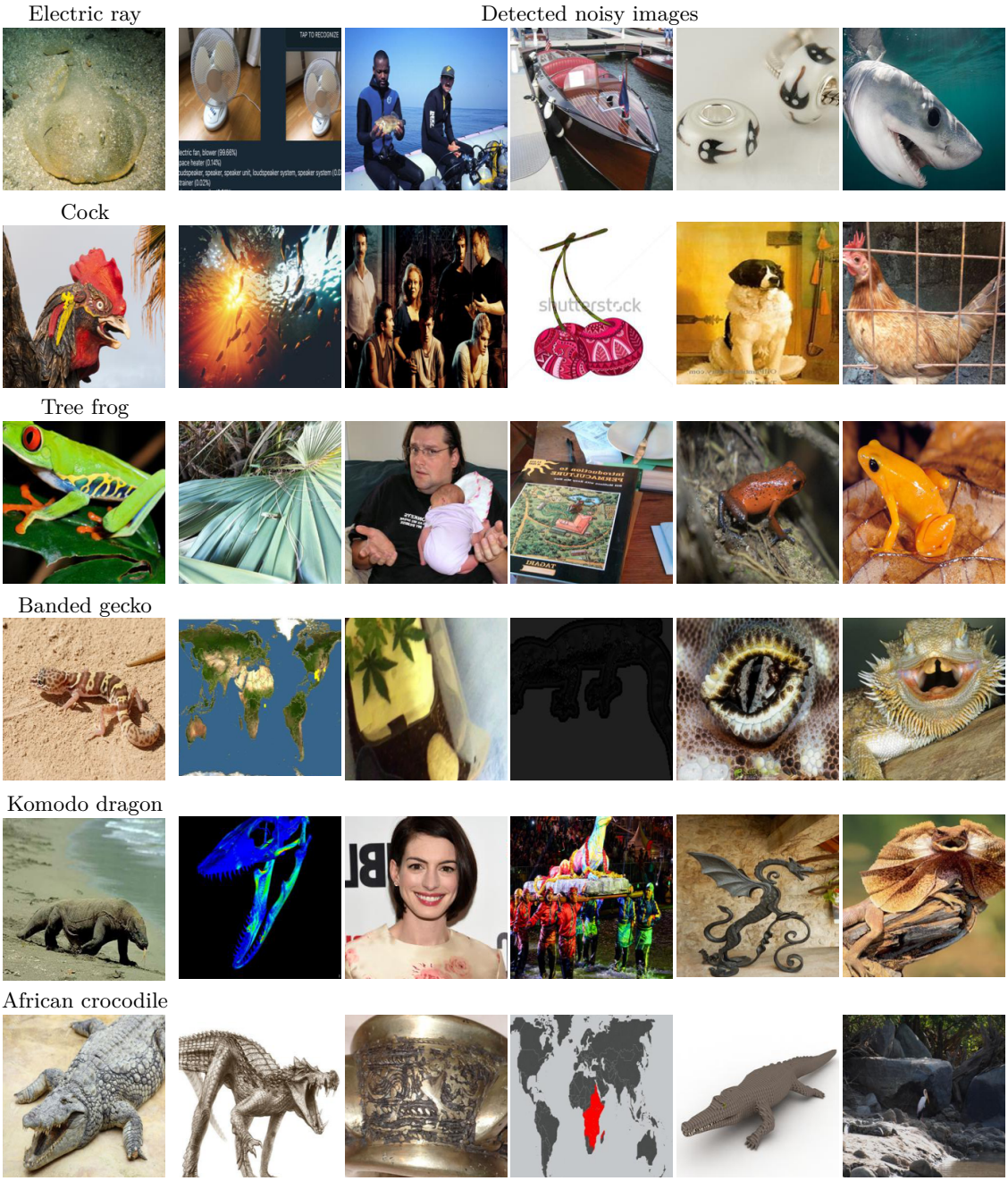

Figure 9. Detected noisy images in WebVision. First column: example of clean image for the class. Second to sixth column: detected noisy images.