Confidence-based Reliable Learning under Dual Noises

Anonymous Author(s)
Affiliation
Address
email

Abstract

Deep neural networks (DNNs) have achieved remarkable success in a variety of computer vision tasks, where massive labeled images are routinely required for model optimization. Yet, the data collected from the open world are unavoidably polluted by noise, which may significantly undermine the efficacy of the learned models. Various attempts have been made to reliably train DNNs under data noise, but they separately account for either the noise existing in the labels or that existing in the images. A naive combination of the two lines of works would suffer from the limitations in both sides, and miss the opportunities to handle the two kinds of noise in parallel. This works provides a first, unified framework for reliable learning under the joint (image, label)-noise. Technically, we develop a confidence-based sample filter to progressively filter out noisy data without the need of pre-specifying noise ratio. Then, we penalize the model uncertainty of the detected noisy data instead of letting the model continue over-fitting the misleading information in them. Experiment results on various challenging synthetic and real-world noisy datasets verify that the proposed method can outperform competing baselines in the aspect of classification performance.

1 Introduction

Deep Neural Networks (DNNs) have obtained great success in a wide spectrum of computer vision applications [26, 40, 19, 18], especially when a large volume of carefully-annotated low-distortion images are available. However, the images collected from the wild in real-world tasks unavoidably have noise in the images themselves (e.g., image corruptions [20] and background noise [42]) or the associated labels [35], termed as image noise (*x-noise*) and label noise (*y-noise*) respectively. Previous investigations show that the DNNs naively trained under *y-noise* [2, 49] or *x-noise* [11, 50] would suffer from detrimental over-fitting issues, thus exhibit poor generalization performance and serious over-confidence.

There has been a large body of attempts towards dealing with data noise, but they mainly focus on a limited setting, where the noise only exists in either the label (i.e., noisy labels) [35, 1, 31, 8] or the image [13, 27, 47]. Moreover, the techniques for handling *x-noise* suffer from non-trivial limitations. For example, most image denoising methods require well-preserved image texture [12], which cannot be satisfied for images that are globally blurred (see Fig. 5 in Appendix); alternative image Super-Resolution (SR) solutions are usually computationally expensive [45]. It is hard for the existing approaches to exhaustively deal with the rich variety of noise in dual noises setting (i.e., the joint *(x,y)-noise*), which raises the requirement of developing a unified approach.

Compared to deterministic DNNs, uncertainty-based deep models (e.g., Bayesian Neural Networks (BNNs) [3] and *deep ensemble* [25]) reason about the uncertainty and hence have the potential to mitigate the over-fitting to noisy data. Empowered by this insight, we first perform a systematical

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

investigation on leveraging uncertainty-based deep models to cope with dual noises (i.e., the joint *(x,y)-noise*). We observe that, despite with less over-fitting, the uncertainty-based deep models may still suffer from the bias in the noisy data and yield compromising results.

To further ameliorate the pathologies induced by data noise and achieve reliable learning, we propose a novel workflow for the learning of uncertainty-based deep models under dual noises. Firstly, inspired by the recent success of using predictive confidence to detect the out-of-distribution data [21], we propose to detect both the noisy images and the noisy labels by the predictive confidence produced by uncertainty-based deep models. Concretely, we use the predictive probability corresponding to the label (i.e., label confidence) to filter out the samples with *y-noise*, and use the maximum confidence to filter out the samples with *x-noise*. After doing so, we propose to penalize the uncertainty [23] of the detected noisy data to make use of the valuable information inside the images without reliance on the misleading supervisory information.

Given the merits of *deep ensemble* [25] for providing calibrated confidence and uncertainty under distribution shift revealed by related works [36] and our empirical study, we opt to place our workflow on deep ensemble to establish a strong learning approach under dual noises. With deep ensemble incorporated, our whole method is easily implementable and scalable. We clarify that the developed strategies for handling dual noises are applicable to other uncertainty-based deep models like BNNs.

We perform extensive empirical studies to evidence the effectiveness of the proposed method. We first show that the proposed method significantly outperforms competitive baselines on CIFAR-100 and TinyImageNet datasets with different levels of synthetic *(x,y)-noise*. We then verify the superiority of the proposed method on the challenging WebVision benchmark [28] which contains extensive real-world noise. We further provide insightful ablation studies to show the robustness of our approach to hyper-parameters.

2 Related work

Many methods have been proposed to deal with *y-noise* in deep learning. A direct approach is to design the robust loss functions, e.g., the loss function based on the mean absolute error [16] and the symmetric cross-entropy [44, 6]. However, it is challenging to deal with the noisy data with high noise rates. An alternative method is to train on reweighing or selected training examples, e.g., estimating the weight of samples based on meta-learning [17], MentorNet [22] and Co-teaching [39], but designing an effective algorithm or criterion of selecting the samples based on the deterministic DNNs tends to be difficult. Recently, the loss correction approaches are also used to mitigate the over-fitting to noisy labels by assigning a weight to the prediction of the model [38, 1] or by adding a regularization to the loss function [31, 8]. To deal with *x-noise*, image denoising may be a useful technique. [13] assumes a uniform camera blur over the image and then applies a standard deconvolution algorithm to reconstruct the blurry image, but it can only handle those locally-blurred images. [47, 34] propose to use a deep convolutional neural network to capture the characteristics of degradation and restorate blurred images, but they commonly need image pairs (i.e., the label indicates clean or noisy) for training and the supervised information cannot be provided in our setting. Therefore, it is not free to extend the existing works to handle dual noises, and developing new techniques is necessary.

Typically, in machine learning and computer vision, the uncertainty we are concerned about can be classified into two categories: *Epistemic* uncertainty and *Aleatoric* uncertainty, which are also called data uncertainty and model uncertainty [23]. Several uncertainty quantification approaches have been proposed in the literature. A direct approach is to incorporate the uncertainty into the expressive DNNs by performing the rigorous Bayesian inference over the network weights in Bayesian deep learning, e.g., BNNs [3, 30], Monte Carlo (MC) dropout [15] and SWAG [32]. Yet, performing Bayesian inference is often challenging due to the high non-linearity of DNNs. An alternative way is to adapt various distance-aware output layers into DNNs in a non-Bayesian way [43, 29, 33]. However, these methods may suffer from degenerated uncertainty estimates [14] due to the limited assumption. *Deep ensemble* [25] is the *de facto* tool for uncertainty quantification, which can produce calibrated confidence and uncertainty [36] by the straightforward ensemble strategy. In this paper, we opt to place our workflow on deep ensemble to establish a robust learning approach under dual noises.

(a) *y-noise*: wrong annotations. (b) *x-noise* I: corrupted images. (c) *x-noise* II: background noise.

Figure 1: An illustration of *y-noise* and *x-noise*.

3 Preliminaries and problem setting

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ denote a collection of image-label pairs, with $x_i \in \mathbb{R}^d$ and $y_i \in \{1, 2, ..., C\}$ as the image and the label respectively. We can routinely deploy a θ-parameterized classifier (e.g., a DNN) $f_\theta : \mathbb{R}^d \to \Delta^C$ for data fitting, where Δ^C is the probability simplex over C classes. In other words, the classifier defines a probability distribution $p_\theta(y|x) = p(y|f_\theta(x))$. Typically, we minimize the cross-entropy loss, i.e., perform maximum likelihood estimation (MLE), to train the model:

$$\min_\theta \ell(\theta; \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} -\log\left(f_\theta\left(x_i\right)\left[y_i\right]\right), \tag{1}$$

where $f_\theta(x_i)[y_i]$ refers to the y_i-th element of the vector $f_\theta(x_i)$. We can also add an L2 penalty on weights $||\theta||_2^2$ to the above objective to achieve maximum a posteriori (MAP) estimation.

To enable the characterization of uncertainty, Bayesian neural networks (BNNs) place a prior distribution over DNNs weights $p(\theta)$, and perform Bayesian inference to find the posterior distribution $p(\theta|\mathcal{D})$ instead of performing MLE or MAP estimation as in the deterministic DNNs. Such an uncertainty-aware modeling can give rise to a more calibrated predictive distribution.

3.1 The setting of learning under noise

In practice, the collected dataset may suffer from heterogeneous *noise*. A typical assumption on data noise is that there are systematical errors in the annotations, i.e., the label noise (*y-noise*). For example, an image of lion may be annotated as "tiger" as shown in Fig. 1a. Tremendous effort has been devoted to handling symmetric, asymmetric, or even instance-dependent *y-noise* [37, 44, 46]. However, in practice, the noise may exist in not only the annotations but also the images themselves (i.e., *x-noise*), casting new challenges for the deep learning models in the real world.

Common *x-noise* includes image corruptions [20] like distortion, blur, compression, etc. (see Fig. 1b). The *x-noise* may also stem from the inherent ambiguity of the image (see Fig. 1c), which is termed as background noise by the previous work [42]. We use *x-noise* I and *x-noise* II to refer to the aforementioned two types of *x-noise* for short. The *x-noise* results in low-quality or even incomplete observations and may cause over-fitting and bias the model. The existing works for dealing with *x-noise* mainly focus on image corruptions (e.g., image denoising and Super-Resolution (SR)), and often require some specific assumptions [12] or expensive computational resources [45]. Therefore, there are still barriers for them to deal with the real-world image noise [12], especially for the background noise.

In this paper, we focus on the learning under dual noises (i.e., the joint *(x,y)-noise*), a more general and more challenging setting than learning under only *x-noise* [20, 42] or *y-noise* [37, 46]. A naive combination of the two lines of works would suffer from the limitations in both sides, and miss the opportunities to handle x-noise and y-noise in parallel. To address this challenge, we need to develop a unified and reliable learning strategy to avoid over-fitting dual noises.

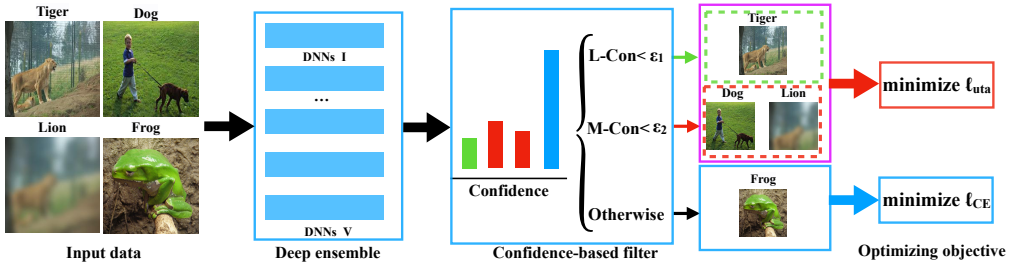


Figure 2: Overview of the proposed method. Given training data with *(x,y)-noise*, the proposed method first distinguishes noisy samples from clean samples using the *confidence-based sample filter*. Then, we can minimize the standard cross-entropy loss for clean data but minimize model uncertainty for noisy data in the framework. ℓ_{uta} represents the loss function of uncertainty penalty.

4 Methodology

Uncertainty-based deep models can potentially mitigate the over-fitting to noisy data due to the inherent characterization of uncertainty. We have conducted a thorough empirical study on using uncertainty-based deep models like BNNs and deep ensemble to handle dual noises (see Appendix A). We found that uncertainty-based models can better alleviate over-fitting than deterministic DNNs. However, these models can still suffer from the bias in the noisy data and yield compromising results. As a result, we propose two strategies to further promote the effectiveness of uncertainty-based deep models for handling dual noises. We place our following discussion upon deep ensemble, one of the best uncertain-based deep models revealed by pioneering works [36] and our study, and clarify that our strategies are compatible with other backbones like BNNs.

We first briefly review deep ensemble. Concretely, a deep ensemble consists of M randomly initialized, individually trained DNNs $\{f_{\theta_m}\}_{m=1}^{M}$, and makes predictions by uniform voting:

$$\frac{1}{M}\sum_{m=1}^{M} f_{\theta_m}(x). \tag{2}$$

As shown, deep ensemble is easy to implement and flexible, which makes our approach enjoys good practicability and scalability.

In the following, we discuss how to construct a confidence-based sample filter to progressively filter out noisy samples, and how to excavate valuable information from detected noisy data. We illustrate our method in Fig. 2.

4.1 The confidence-based sample filter

Distinct from leveraging complicated strategies for noise detection in previous works [1, 31], we propose a simple confidence-based sample filter to filter out *x-noise* and *y-noise* in parallel.

Filtering out *y-noise* using the Label confidence (L-Con). Specifically, we first use the predictive probability corresponding to the label y (i.e., the label confidence) to distinguish the data with *y-noise* from the others. In the case of deep ensemble, the label confidence can be simply estimated by:

$$\textbf{L-Con}(x) = \frac{1}{M}\sum_{m=1}^{M} f_{\theta_m}(x)[y]. \tag{3}$$

Intuitively, **L-Con** reflects how confident the model is for the current input w.r.t. the label. Our hypothesis is that our model tends to yield low **L-Con** for the training data with *y-noise* yet yield high **L-Con** for the others. We empirically corroborate this in Fig. 4 in Appendix. As shown, the data with *y-noise* can be accurately distinguished from the clean data by **L-Con**. More importantly, the **L-Con** of the data with *y-noise* is not mixed up with that of the clean data even at the later training phase (see Fig. 4b).

Filtering out *x-noise* using the Maximum confidence (M-Con). We then move to the detection of *x-noise*. Inspired by the success of using the maximum confidence for out-of-distribution detection [21],

we utilize the maximum confidence to detect the training data with *x-noise*. The maximum confidence of deep ensemble takes the form of

$$\textbf{M-Con}(x) = \max_j \left(\frac{1}{M} \sum_{m=1}^{M} f_{\theta_m}(x) \right)[j]. \tag{4}$$

Why M-Con is effective in detecting *x-noise*? In fact, there is an inherent connection between **M-Con** and data uncertainty (i.e., aleatoric uncertainty). Recalling the explanation in [23, 10], the data/aleatoric uncertainty represents the magnitude of the inherent data noise (e.g., sensor noise), and can be estimated by

$$\text{Ale}(x) = \mathbb{E}_{p(\theta|\mathcal{D})}(\mathcal{H}(p(y|x,\theta)) = \frac{1}{M} \sum_{m=1}^{M} \mathcal{H}(p_{\theta_m}(y|x,\theta)),$$

where $\mathcal{H}$ is the Shannon entropy, and it can be directly estimated by

$$H[p(y|x)] = -\sum_{c=1}^{C} (f_\theta(x)[c])(\log f_\theta(x)[c]), \tag{5}$$

where C is the number of classes. When the model is confident in its prediction (i.e., **M-Con** is high), it yields a sharp predictive distribution centered on one of the corners of the simplex. In contrast, when the model is not confident in its prediction (i.e., **M-Con** is low), it yields a flat predictive distribution scattered in every direction of the simplex, which corresponds to a high data uncertainty. There is evidence showing that the data uncertainty grows as the quality of the input image degrades [5], so **M-Con** is effective in detecting the noisy data with *x-noise*.

How to filter? We propose a simple yet efficient sample filter based on **L-Con** and **M-Con**. To be specific, we first assign different weights for different data according to the value of **L-Con**,

$$w_i^l = \begin{cases} 0, & \text{if } \textbf{L-Con}(x_i) \leq \epsilon_1 \\ 1, & \text{otherwise,} \end{cases} \tag{6}$$

where ϵ_1 is the threshold for filtering out *y-noise*, and w_i^l indicates whether the label of input sample is noisy (w_i^l = 0) or clean (w_i^l = 1). Likewise, we can also filter out the samples with *x-noise* according to the value of **M-Con**:

$$w_i^k = \begin{cases} 0, & \text{if } \textbf{M-Con}(x_i) \leq \epsilon_2 \\ 1, & \text{otherwise.} \end{cases} \tag{7}$$

ϵ_2 is the threshold to decide whether the input sample is clean (w_i^k = 1) or not (w_i^k = 0).

After twice filtering, the final sample weight is $w_i^s = w_i^l \times w_i^k$. Generally, we first train the deep ensemble under a high learning rate for some epochs, after which we use the confidence-based sample filter to filter out noisy data at per iteration. The foregoing warm-up can make the sample filter better for distinguish the noisy data from the clean one.

4.2 Uncertainty penalty on noisy data

We first discuss the limitations of the typical learning objectives for dealing with dual noises. Then, we propose an improved learning objective based on model uncertainty.

Limitations of typical learning objectives. After distinguishing the clean samples from the noisy ones, it is necessary to resort to some new learning objectives to drive the model training, since that continuing pushing the model to fit dual noises may exacerbate the over-fitting. Typical strategies like the loss correction technique [38, 1] regard the model predictions as pseudo labels and minimize the following loss

$$\ell(\theta; \mathcal{D}) = -\sum_{i=1}^{N} \left(\alpha_i \log \left(f_\theta \left(x_i \right) [y_i] \right) + \beta_i \sum_{c=1}^{C} f_\theta(x_i)[c] \log(f_\theta(x_i)[c]) \right), \tag{8}$$

where α and β are the weights for clean data and noisy labels.

Nevertheless, it is non-trivial to extent these strategies to dealing with the data with *x-noise*. On the one hand, the existing methods are limited to detecting *y-noise* using the cross-entropy loss. On the other hand, the dirty observations (e.g., the corrupted images) are unhelpful for model learning. The model cannot make reliable predictions for the images with *x-noise*, so incorporating the predictions of these samples into the loss function may be harmful.

The model uncertainty estimation. Fortunately, we notice that deep ensemble can offer high-quality measures of model uncertainty for the input data [25, 36]. By penalizing the model uncertainty of noisy data, we can make our model certain on the training data with *(x,y)-noise*. Specifically, the model uncertainty can be measured by the mutual information between the predictions and the model parameters [10, 41].

$$\underbrace{\mathcal{I}\left[y,\theta|x;\mathcal{D}\right]}_{\text{Model Uncertainty}} = \underbrace{\mathcal{H}\left[\mathbb{E}_{p_{(\theta|\mathcal{D})}}(p(y|x,\theta))\right]}_{\text{Total Uncertainty}} - \underbrace{\mathbb{E}_{p_{(\theta|\mathcal{D})}}[\mathcal{H}(p(y|x,\theta)]}_{\text{Data Uncertainty}},$$

which, in the case of deep ensemble, boils down to

$$\mathcal{I}\left[y,\theta|x;\mathcal{D}\right] \approx \mathcal{H}[\frac{1}{M}\sum_{m=1}^{M} p_{\theta_m}(y|x)] - \frac{1}{M}\sum_{m=1}^{M}\mathcal{H}[p_{\theta_m}(y|x)]. \tag{9}$$

The proposed learning objective. Specifically, we optimize the following loss for each ensemble member in deep ensemble:

$$\min_{\theta_m} \ell(\theta_m;\mathcal{D}) = \begin{cases} \sum_{i=1}^{N} -\log\left(f_{\theta_m}\left(x_i\right)\left[y_i\right]\right), & \text{if } w_i^s = 1 \\ \sum_{i=1}^{N} \mathcal{I}(y,\theta|x_i,\mathcal{D}), & \text{if } w_i^s = 0 \end{cases} \tag{10}$$

where w_i^s is the weight of each sample. Namely, we minimize the standard cross-entropy loss for clean data, but minimize the model uncertainty for noisy data. Intuitively, the former allows the model to constantly learn useful information when the labels and images are reliable. The latter enables the model to explore the valuable information inside the noisy data, while preventing the model from being misled by the harmful supervisory information. We detail the whole process of the proposed method in Algorithm 1.

Algorithm 1: Training DNNs under *(x,y)-noise*

Input: Training noisy dataset $\mathcal{D}$, number of networks M for ensemble, **L-Con** threshold ϵ_1, **M-Con** threshold ϵ_2

```
Initialize M networks f_{θ_1}, ··· , f_{θ_M};
for m = 1 : M do
    θ^(m) ← WarmUp(D, θ^(m));
end
while e < MaxEpoch do
    for Mini-batch B in D do
        Compute L-Con and M-Con using equation 3 and 4;
        Determine weights w_i^l and w_i^k following thresholding rule 6 and 7;
        Update each network f_{θ_m} with loss function
          L(θ_m, B) = Σ_{(x_i,y_i)∈B} (1 − w_i^k w_i^l) I(y_i, θ) + w_i^k w_i^l L_CE(θ_m, B);
    end
    e = e + 1
end
```

5 Experiment

In this section, we first thoroughly evaluate the proposed method on the datasets with synthetic noise and the real-world noisy dataset: WebVision. Furthermore, we ablate the robustness of the proposed method to hyper-parameters in terms of the number of ensembles: M and two thresholds: ϵ_1 and ϵ_2. Besides, we also verify the effectiveness of the uncertainty penalty strategy in ablation studies.

Datasets. The proposed method is first evaluated on two benchmark datasets with synthetic noise: CIFAR-100 [24] and TinyImageNet [24] (the subset of ImageNet[9]), the former consists of 100 classes with 32x32 color images, and the latter has 200 classes with 64x64 color images. Moreover, we validate the effectiveness of the proposed method under more challenging real-world noise on WebVision [28], which contains more than 2.4 million images crawled from the Flickr website and Google Images search.

Implementation details. The synthetic noise contains the common *y-noise* used in [48, 1] and *x-noise* I: the corruption on images. We use the symmetric noise as the synthetic *y-noise*, which is generated by randomly flipping the true label to other possible labels. For *x-noise* I, we randomly apply the challenging "Gaussian Blur", "Fog" and "Contrast" corruption used in [20] to the original images to simulate the real-world image noise. The *x-noise* II (i.e., background noise) commonly exists in web images, thus we also evaluate the proposed method on WebVision dataset. The deep ensemble we used consists of 5 ResNet18 [19] for all datasets. SGD is used to optimize the network with a batch size of 256. More details can be found in Appendix B.

Baselines. The first thing to note is that all methods employ 5 networks for fair comparisons. We compare with two kinds of compared baselines. The first kind contains the single model (Single-CE) and deep ensemble (DE-CE) with the standard cross-entropy loss. The second kind is competitive loss correction technique related to our method, which involves the regularized loss function with dynamic bootstrapping (DYR) [1], the regularized loss function with mixup dynamic bootstrapping (M-DYR) [1] and COnfidence REgularized Sample Sieve (CORES^2) [8]. Besides, we use "Proposed-L (Proposed-M)" to indicate that we only use **L-Con** (**M-Con**) to filter out noisy samples and use "Proposed-LM" to represent the proposed method with **L-Con** and **M-Con** filter. Furthermore, we also consider the pipeline of combining the denoising technique and M-DYR as a compared baseline. However, as shown in Fig. 5 in Appendix, we can observe that existing denoising methods do not restore globally blurred images. As a consequence, a more effective strategy is to filter out low-quality images in this paper rather than restore them.

5.1 Performance under synthetic (x,y)-noise

In this section, we first empirically evaluate the proposed method and other baselines on CIFAR-100 and TinyImageNet with different levels of synthetic *(x,y)-noise*. Afterward, we also compare the proposed method with competitive baselines under the label noise.

Table 1: The comparison of validation accuracy on CIFAR-100 and TinyImageNet with *(x,y)-noise*. "$0.2y + 0.3x$" represents the dataset with 20% *y-noise* and 30% *x-noise* simultaneously.

Alg./Noise rate		0.0	$0.3x$	$0.4x$	$0.2y$+$0.3x$	$0.4y$+$0.3x$
		CIFAR-100 / TinyImageNet				
Single-CE	Best	77.23/61.19	73.62/54.39	72.53/52.53	57.84/43.59	47.76/40.62
	Last	76.44/60.26	72.19/49.03	71.95/49.95	57.39/36.81	41.39/22.62
DE-CE	Best	79.13/63.62	77.07/60.03	76.12/59.94	66.50/50.03	54.90/46.36
	Last	77.01/61.28	76.14/59.51	74.98/59.05	65.24/46.21	53.91/41.27
DYR [1]	Best	78.64/**65.14**	73.64/60.74	71.68/59.20	62.54/52.14	50.54/43.94
	Last	78.02/63.97	73.07/59.25	71.13/58.01	60.59/50.67	49.21/40.89
M-DYR [1]	Best	75.38/62.32	75.11/60.70	73.86/59.45	72.38/52.14	64.07/50.50
	Last	74.91/61.04	74.28/58.44	72.41/57.21	70.69/50.04	62.34/48.02
CORES^2 [8]	Best	76.76/59.74	73.15/57.22	72.04/55.67	63.06/46.40	51.98/44.55
	Last	76.22/59.14	73.01/56.35	71.98/54.41	62.51/44.91	51.11/43.20
Proposed-L	Best	80.44/64.07	77.01/60.68	76.06/59.54	71.05/57.62	63.04/51.41
	Last	79.03/63.21	76.58/59.99	75.08/58.62	69.91/56.31	61.97/50.23
Proposed-M	Best	80.61/64.37	77.89/**61.06**	77.51/**60.51**	-/-	-/-
	Last	79.39/64.01	77.02/**60.26**	77.19/**59.34**	-/-	-/-
Proposed-LM	Best	**80.98**/64.58	**77.92**/61.01	**77.53**/60.12	**72.78/58.75**	**66.61/52.35**
	Last	**79.71/64.15**	**77.03**/60.23	**77.32**/59.19	**72.48/57.82**	**66.05/51.02**

We evaluate the classification accuracy at the best and last epoch following the setting of [1]. Table 1 presents the results of all methods on CIFAR-100 and TinyImageNet with different rates of *x-noise* and *y-noise*. We can see that the proposed method outperforms other baselines under synthetic *(x,y)-noise* in terms of classification accuracy at the best and last epoch. Especially, the proposed method achieves a remarkable performance improvement comparing other methods under the joint *(x,y)-noise* (i.e., "0.2y+0.3x" and "0.4y+0.3x" in Table 1), which shows the effectiveness of the proposed method to handle dual noises.

Besides, we can observe that Proposed-M outperforms DE-CE under *x-noise* (i.e, "0.3x" and "0.4x"), which shows that the effectiveness of employing **M-Con** to filter out samples with *x-noise*. By contrast, the previous works that focus on the noisy label (i.e., DYR, M-DYR and CORES^2) do not show the superior performance regardless of whether *x-noise* or *(x,y)-noise*, which confirms that they cannot effectively handle dual noises. Moreover, we can notice that the naive deep ensemble with cross-entropy loss (DE-CE) significantly outperforms the single model (Single-CE), confirming that uncertainty-based deep ensemble can prevent the model from over-fitting noisy data. In addition, we can notice that the experimental results exhibit quite close best accuracy and last accuracy, which shows that our method is not easy to over-fit noisy data and can achieve stable and robust learning.

To verify the effectiveness of the proposed method under label noise (i.e., *y-noise*), we also compare our method with other baselines on CIFAR-100 and TinyImageNet with different levels of synthetic *y-noise* in Appendix C. The experimental results show that the proposed method significantly outperforms competitive methods for noisy labels. Specifically, "Proposed-L" also outperforms or is close to the best results of other baselines. We discuss more details in Appendix C.

5.2 Performance on the real-world noisy dataset

Furthermore, we verify the generalization performance of the proposed method on a large real-world noisy dataset: WebVision. Since the dataset is too big, for quick experiments, we compare all methods on the first 50 classes of the Google image subset and use the resized images following previous works [22, 7]. Besides, we test the trained model of all methods on the human-annotated WebVision validation set and the ILSVRC12 validation set [9]. Table 2 lists the experimental results. As we can see, the proposed method significantly outperforms other baselines not only on the WebVision validation set but also on the ILSVRC12 validation set for the real-world noisy dataset, which shows the superiority of our method is also effective to the real-world noisy dataset.

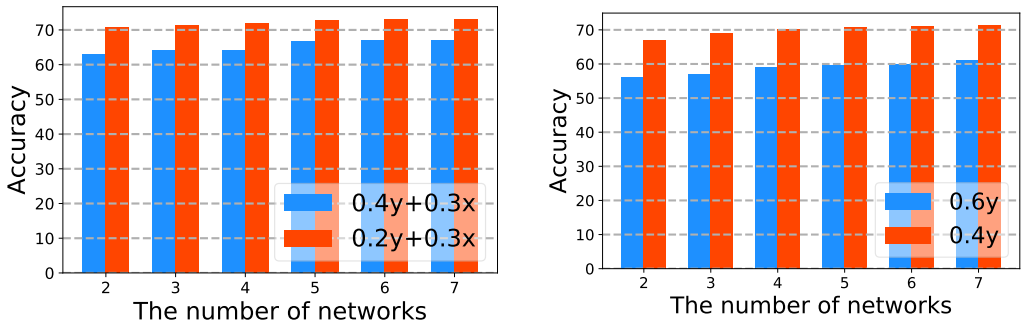
Table 2: The comparison of validation accuracy on ImageNet ILSVRC12 and WebVision validation set. The number outside (inside) the parentheses denotes top-1 (top-5) accuracy.

Val./Methods		DYR	M-DYR	CORES^2	DE-CE	Proposed-LM
WebVision	Best	69.48 (83.21)	72.36 (87.40)	70.56 (87.56)	73.76 (88.13)	**76.68 (91.32)**
	Last	68.53 (82.42)	72.01 (87.15)	69.52 (87.02)	73.22 (87.98)	**76.52 (91.22)**
ILSVRC12	Best	67.32 (89.76)	68.52 (86.36)	64.12 (86.36)	67.64 (88.73)	**71.40 (90.88)**
	Last	66.59 (88.98)	68.33 (86.21)	63.23 (85.44)	67.31 (88.26)	**71.26 (90.70)**

5.3 Ablation studies

Empirical effects of the number of networks M. The number of networks for deep ensemble is a crucial hyper-parameter. Empirically, the more networks for deep ensemble, the more powerful performance can achieve. However, assembling a large number of networks often requires high memory and computational costs. Hence, we need to make an appropriate trade-off between the performance and the computational cost. Fig. 3 demonstrates the performance of the proposed method corresponding to the different numbers of networks under different levels of *(x,y)-noise* on CIFAR-100. We can see that even a small number of networks can not overly drop the performance. When the number of networks is greater than 4, the proposed method can almost achieve the best performance, so an ensemble of 5 networks is enough for our method.

Empirical effects of thresholds of confidence-based sample filter. Moreover, we analyze the effects of hyper-parameters: ϵ_1 and ϵ_2 of the proposed confidence-based sample filter on the predictive performance. For the threshold of **M-Con**, we use a soft threshold to filter out the training data with *x-noise* after per iteration (i.e., the training data with minimum $\epsilon_2\%$ **M-Con** is filtered out), which is more effective than the hard threshold through empirical studies. For the threshold of **L-Con**, the



(a) The performance under different levels of (x,y)-noise. (b) The performance under different levels of y-noise.

Figure 3: Effects of different numbers of networks on the performance of the proposed method on CIFAR-100.

hard threshold is more appropriate according to the empirical results in Fig. 4. Table 3 reports the comparison results of different thresholds on CIFAR-100. We can observe that the performance of the proposed method is not sensitive to ϵ_1 and ϵ_2, which can achieve superior performance within a certain range of thresholds. In summary, our method indeed shows the effectiveness and practicability of dealing with noisy data, which does not rely on time-consuming hyper-parameters tuning.

Table 3: The comparison of validation accuracy under different ϵ_1 and ϵ_2 on CIFAR-100 with different levels of *(x,y)-noise*.

ϵ_1 (10^{-2})		1.5	2.0	2.5	3.0	3.5	ϵ_2 (%)	2.0	3.0	4.0	5.0	6.0
$0.4y$+$0.3x$	Best	66.01	66.61	66.57	66.34	**66.58**	Best	64.92	65.81	66.02	**66.72**	66.11
	Last	65.63	66.05	66.04	**66.10**	66.01	Last	64.43	64.52	65.59	**66.09**	66.04
$0.2y$+$0.3x$	Best	71.92	**72.78**	72.61	72.69	72.71	Best	71.93	72.51	72.76	**72.93**	72.88
	Last	71.22	**72.48**	72.24	72.45	72.44	Last	71.85	72.33	72.24	**72.59**	72.49
$0.6y$	Best	57.85	**59.65**	59.52	58.59	57.94	Best	58.20	58.92	58.99	59.07	**59.61**
	Last	55.96	**55.53**	55.26	55.06	55.03	Last	57.01	56.63	56.17	**56.44**	55.21
$0.4y$	Best	70.63	**70.77**	70.28	70.11	69.88	Best	69.82	69.98	**70.81**	70.62	70.59
	Last	68.27	**68.83**	68.19	67.63	67.46	Last	67.89	68.07	68.94	68.71	**68.72**

Effects of uncertainty penalty in the proposed learning objective. To verify the effectiveness of uncertainty penalty in Eqn. (10), we report the performance of the proposed method without uncertainty penalty and the gap with "Proposed-LM" in Table 4. We observe that the validation accuracy is lower than the complete workflow on both *y-noise* and *(x,y)-noise*, which clarify that the effectiveness of the uncertainty penalty strategy.

Table 4: The best accuracy on CIFAR-100 and TinyImageNet with *(x,y)*-noise.

Noise rate	$0.4y$	$0.6y$	$0.2y$+$0.3x$	$0.4y$+$0.3x$
	CIFAR-100 / TinyImageNet			
Best Acc	67.92/54.23	57.33/41.52	70.81/55.34	63.02/49.68
Gaps	2.85/1.98	2.17/3.13	1.97/3.41	3.59/2.67

6 Conclusions

This work first introduces the more challenging and closer to real-world noise setting and then performs a systematical investigation on using uncertainty-based models under dual noises (i.e., the joint *(x,y)-noise*). We find that merely employing an uncertainty-based model is not enough and furthermore propose a novel workflow for the learning of uncertainty-based deep models. Concretely, we present the efficient and practical confidence-based sample filter to distinguish noisy data from clean data progressively. After doing so, we propose to penalize the model uncertainty of noisy data without reliance on the misleading supervisory information. Empirically, the proposed method significantly outperforms the competitive baselines on CIFAR-100 and TinyImageNet with synthetic *(x,y)-noise* and the real-world noisy dataset. We further evaluate the robustness of hyper-parameters in our method, which shows that the proposed method is not sensitive to crucial hyper-parameters. In the future, this work may promote more approaches to deal with dual noises in more tasks.

References

[1] Eric Arazo, Diego Ortego, Paul Albert, Noel O'Connor, and Kevin McGuinness. Unsupervised label noise modeling and loss correction. In *International Conference on Machine Learning*, pages 312–321. PMLR, 2019.

[2] Devansh Arpit, Stanisaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pages 233–242. PMLR, 2017.

[3] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, pages 1613–1622, 2015.

[4] Guillermo Carbajal, Patricia Vitoria, Mauricio Delbracio, Pablo Musé, and José Lezama. Non-uniform motion blur kernel estimation via adaptive decomposition. *arXiv e-prints*, pages arXiv–2102, 2021.

[5] Jie Chang, Zhonghao Lan, Changmao Cheng, and Yichen Wei. Data uncertainty learning in face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5710–5719, 2020.

[6] Nontawat Charoenphakdee, Jongyeong Lee, and Masashi Sugiyama. On symmetric losses for learning from corrupted labels. In *International Conference on Machine Learning*, pages 961–970. PMLR, 2019.

[7] Pengfei Chen, Ben Ben Liao, Guangyong Chen, and Shengyu Zhang. Understanding and utilizing deep neural networks trained with noisy labels. In *International Conference on Machine Learning*, pages 1062–1070. PMLR, 2019.

[8] Hao Cheng, Zhaowei Zhu, Xingyu Li, Yifei Gong, Xing Sun, and Yang Liu. Learning with instance-dependent label noise: A sample sieve approach. In *International Conference on Learning Representations*, 2020.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Stefan Depeweg, Jose-Miguel Hernandez-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *International Conference on Machine Learning*, pages 1184–1193. PMLR, 2018.

[11] Samuel Dodge and Lina Karam. Understanding how image quality affects deep neural networks. In *2016 eighth international conference on quality of multimedia experience (QoMEX)*, pages 1–6. IEEE, 2016.

[12] Linwei Fan, Fan Zhang, Hui Fan, and Caiming Zhang. Brief review of image denoising techniques. *Visual Computing for Industry, Biomedicine, and Art*, 2(1):1–12, 2019.

[13] Rob Fergus, Barun Singh, Aaron Hertzmann, Sam T Roweis, and William T Freeman. Removing camera shake from a single photograph. In *ACM SIGGRAPH 2006 Papers*, pages 787–794. 2006.

[14] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

[15] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[16] Aritra Ghosh, Himanshu Kumar, and PS Sastry. Robust loss functions under label noise for deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

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

[18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[20] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2018.

[21] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2016.

[22] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313. PMLR, 2018.

[23] Alex Kendall and Yarin Gal. What uncertainties do we need in Bayesian deep learning for computer vision? *Advances in neural information processing systems*, pages 5574–5584, 2017.

[24] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[25] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, pages 6402–6413, 2017.

[26] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[27] Anat Levin, Yair Weiss, Fredo Durand, and William T Freeman. Understanding and evaluating blind deconvolution algorithms. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1964–1971. IEEE, 2009.

[28] Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017.

[29] Jeremiah Zhe Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax-Weiss, and Balaji Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *Advances in Neural Information Processing Systems*, 33, 2020.

[30] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose Bayesian inference algorithm. In *Advances in Neural Information Processing Systems*, pages 2378–2386, 2016.

[31] Yang Liu and Hongyi Guo. Peer loss functions: Learning from noisy labels without knowing noise rates. In *International Conference on Machine Learning*, pages 6226–6236. PMLR, 2020.

[32] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, pages 13153–13164, 2019.

[33] Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7047–7058, 2018.

[34] Xiaojiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. *Advances in neural information processing systems*, 29:2802–2810, 2016.

[35] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. *Advances in neural information processing systems*, 26:1196–1204, 2013.

[36] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in Neural Information Processing Systems*, 32:13991–14002, 2019.

[37] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1944–1952, 2017.

[38] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *International Conference on Learning Representations*, 2015.

[39] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*, pages 4334–4343. PMLR, 2018.

[40] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.

[41] Lewis Smith and Yarin Gal. Understanding measures of uncertainty for adversarial example detection. In *AUAI*, 2018.

[42] Yi Tu, Li Niu, Junjie Chen, Dawei Cheng, and Liqing Zhang. Learning from web data with self-organizing memory module. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12846–12855, 2020.

[43] Joost Van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. In *International Conference on Machine Learning*, pages 9690–9700. PMLR, 2020.

[44] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 322–330, 2019.

[45] Zhihao Wang, Jian Chen, and Steven CH Hoi. Deep learning for image super-resolution: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[46] Xiaobo Xia, Tongliang Liu, Bo Han, Nannan Wang, Mingming Gong, Haifeng Liu, Gang Niu, Dacheng Tao, and Masashi Sugiyama. Part-dependent label noise: Towards instance-dependent label noise. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[47] Li Xu, Jimmy SJ Ren, Ce Liu, and Jiaya Jia. Deep convolutional neural network for image deconvolution. In *Proceedings of the 27th International Conference on Neural Information Processing Systems-Volume 1*, pages 1790–1798, 2014.

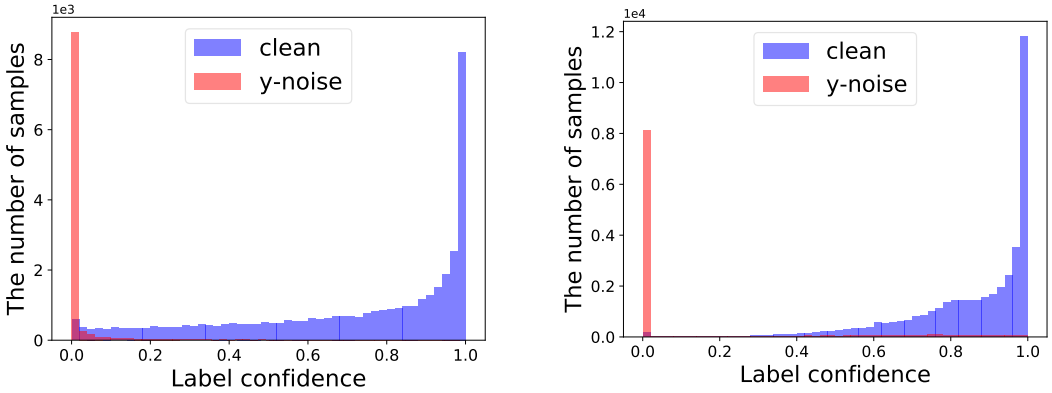
[48] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization (2016). In *International Conference on Learning Representations*, 2017.

[49] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. In *International Conference on Learning Representations*, 2019.

[50] Yiren Zhou, Sibo Song, and Ngai-Man Cheung. On classification of distorted images with deep convolutional neural networks. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1213–1217. IEEE, 2017.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No] We have not yet found the potential negative impact of our work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No] The official websites containing the datasets have already provided the corresponding license.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] The datasets used are all public in this paper.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]



(a) Histogram of **L-Con** at 40th epoch.

(b) Histogram of **L-Con** at 100th epoch.

Figure 4: **L-Con** distributions at training phase on CIFAR-100 with 40% *y-noise*.

A The performance of different uncertainty-based models under dual noises

Table 5: The comparison of validation accuracy on CIFAR-100 with the joint *(x,y)-noise*. "$0.2y+0.3x$" represents the dataset with 20% *y-noise* and 30% *x-noise* simultaneously.

Methods/Val.		$0.2y$+$0.3x$.	$0.4y$+$0.3x$
Single-CE	Best	57.84	47.76
	Last	57.39	41.39
BNNs	Best	62.25	51.20
	Last	61.96	50.68
SNGP	Best	59.92	49.27
	Last	59.22	49.09
DE-CE	Best	**66.50**	**54.90**
	Last	**65.24**	**53.91**

We also explore more uncertainty-based models in addition to deep ensemble to fit the noisy data with dual noises, e.g., Bayesian neural networks (BNNs) with mean-field variational inference (MFVI) and Spectral-normalized Neural Gaussian Process (SNGP) [43]. We do not consider the uncertainty-based models (e.g., Monte Carlo (MC) dropout [15], DUQ [29] and Prior Network [33]) that can not explicitly model uncertainty at the training phase in our experiments because the lack of uncertainty can not alleviate over-fitting during the training time. Table 5 presents the classification accuracy of uncertainty-based models and deterministic DNNs on CIFAR-100 with dual noises. We can see that uncertainty-based models can better alleviate over-fitting than deterministic DNNs. Especially, deep ensemble used in this paper can achieve the best performance compared to BNNs and SNGP, so we opt to place our workflow on the well-evaluated deep ensemble to establish a strong learning approach under dual noises.

B Experiment details

B.1 Preprocessing

All images are normalized and augmented by random horizontal flipping. For CIFAR-100, we use the standard 32×32 random cropping after zero-padding with 4 pixels on each side. For TinyImageNet, we use randomly crop a patch of size 56×56 from each image. For WebVision, we first resize each image to make the size as 320. Then we use the standard data augmentation, randomly crop a patch of size 299×299 from each image, and apply horizontal random flipping.

(a) The corrupted image using the Gaussian blur.

(b) The restored image using the mean filter in *OpenCV*.

(c) The restored image using the latest denoising technique based on NNs [4].

Figure 5: Blurred and restored images using different image denoising methods.

B.2 Optimizer and hyper-parameters

SGD with momentum (0.9) and weight decay 3×10^{-4} is used in all experiments. For the setting of the thresholds ϵ_1 and ϵ_2, we recommend performing a grid search for $\epsilon_1 \in \{0.015, 0.020, 0.025, 0.030, 0.035\}$ and $\epsilon_2 \in \{1\%, 2\%, 3\%, 4\%, 5\%, 6\%\}$ to achieve the better performance. In the experiments, we set $\epsilon_1 = 0.020$ and $\epsilon_2 = 5\%$ for CIFAR-100 and TinyImageNet.

B.3 Network

Five networks with ResNet-18 are trained from scratch using PyTorch 1.9.0. for all experiments. Default PyTorch initialization is used on all layers. It is noteworthy that we need to use the small convolution with 3×3 kernel in the downsampling layer for CIFAR-100 and TinyImageNet.

B.4 Warm-up

The model warm-up can help better separate noisy data and clean data. We start training the model with high learning rates and standard cross-entropy loss in experiments. Specifically, our method uses the learning rate of 0.2 for the first 35 epochs for CIFAR-100 and TinyImageNet. For WebVision, we use the learning rate of 0.2 for the first 40 epochs.

B.5 Training schedule

For CIFAR-100 and TinyImageNet, training for 250 epochs in total, and we reduce the initial learning rate (0.2) by a factor of 2.5 after 35, 80, 120, 150 and 180 epochs. For WebVision, we train the model for 130 epochs and reduce the initial learning rate (0.1) by a factor of 10 after 80 and 105 epochs.

C Performance under synthetic y-noise

We report the results of all methods only under label noise in order to directly compare the proposed method with previous works focusing on this setting. Table 6 presents the results on CIFAR-100 and TinyImageNet with different levels of *y-noise*, which shows that the proposed method significantly outperforms competitive methods for noisy labels (i.e., DYR, M-DYR, and CORES^2). The results verify that our method is also suitable for scenarios that only involve *y-noise*. In particular, the proposed methods (i.e., no matter Proposed-L or Proposed-LM) also exhibit superior performance on clean data(i.e., 0% noise). The result illustrates the proposed sample filtering and learning strategy is robust and can not bias the learning of the model on overall clean datasets, which is not enjoyed by other methods.

Table 6: The comparison of validation accuracy on CIFAR-100 and TinyImageNet with *y-noise*.

Alg./Noise rate		0.0	0.1	0.2	0.4	0.6
		CIFAR-100 / TinyImageNet				
DE-CE	Best	79.13/63.62	75.30/60.03	71.19/55.82	60.65/47.89	51.26/39.14
	Last	77.01/61.28	75.16/58.06	70.84/53.02	59.16/41.00	42.74/32.36
DYR [1]	Best	78.64/**65.14**	73.76/60.04	68.47/56.33	58.43/47.85	46.02/37.19
	Last	78.02/63.97	73.13/58.13	67.31/54.28	57.06/45.72	44.91/35.86
M-DYR [1]	Best	75.38/62.32	75.43/60.40	75.18/**59.59**	69.43/54.59	59.48/42.06
	Last	74.91/61.04	75.12/59.11	74.69/**58.25**	68.73/52.77	56.07/41.26
CORES^2 [8]	Best	76.76/59.74	71.79/57.15	67.42/54.53	55.18/46.95	42.97/37.17
	Last	76.22/59.14	71.03/57.00	66.62/53.26	54.31/45.39	41.89/36.02
Proposed-L	Best	80.44/64.07	**79.45**/60.25	**75.76**/58.96	69.44/59.02	58.76/43.00
	Last	79.03/63.21	77.15/59.31	74.89/58.87	66.04/**55.98**	57.02/41.54
Proposed-LM	Best	**80.98**/64.58	79.32/**60.73**	75.20/58.65	**70.77/56.21**	**59.52/44.65**
	Last	**79.71/64.15**	**78.81/59.95**	**75.15**/58.01	**68.83**/55.84	**56.26/44.10**

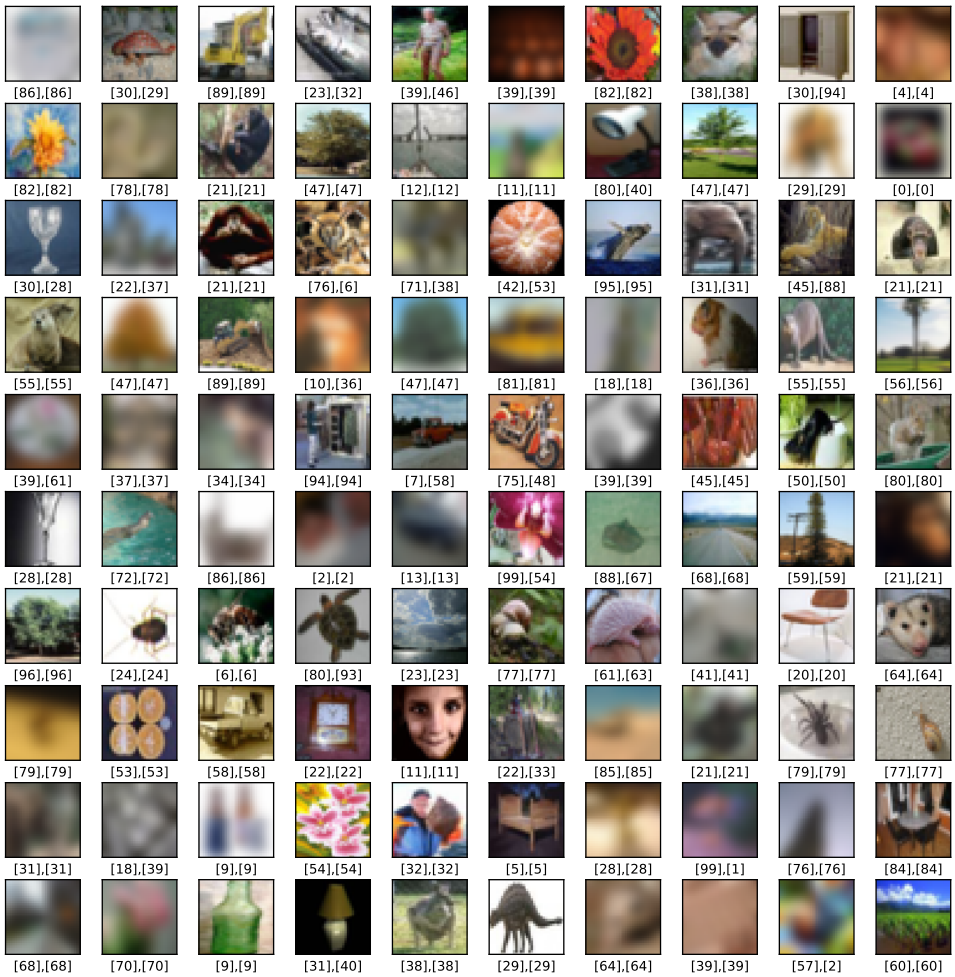


Figure 6: Some random images of CIFAR-100 with dual noises. The two numbers below each image represent the actual label id and the correct label id respectively. If two ids are not identical, it indicates an image with label noise.

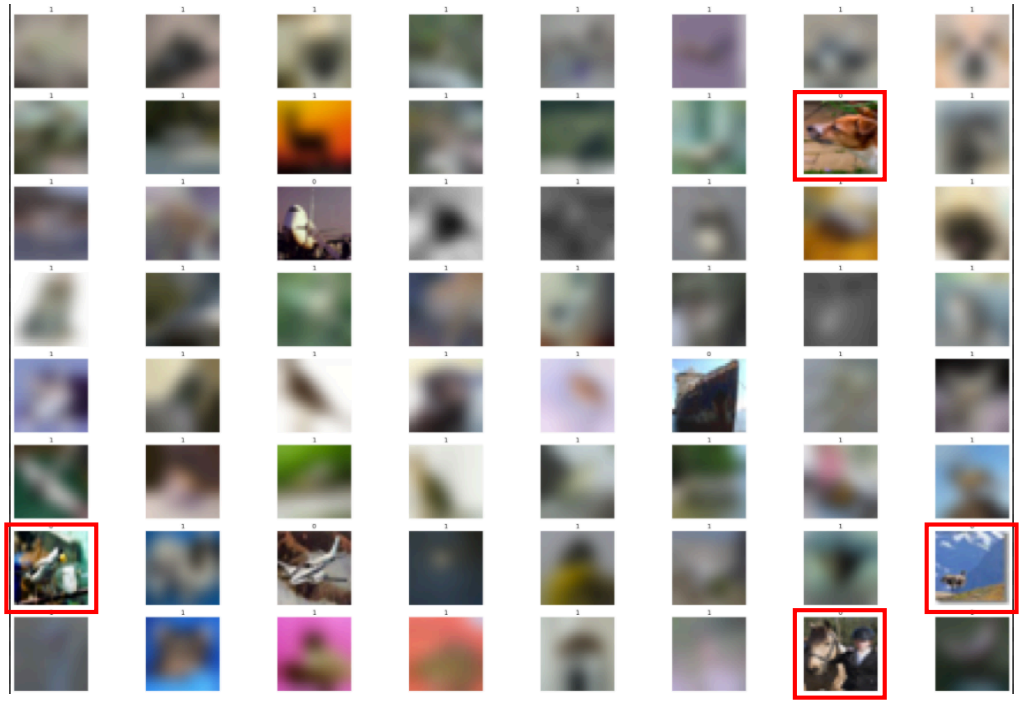


Figure 7: There are some low **M-Cons** corresponding images. Images with red boxes represent hard or dirty samples (e.g., some images contain multiple objects or some images contain background noise).