

PADDLES: Phase-Amplitude Spectrum Disentangled Early Stopping for Learning with Noisy Labels

Huaxi Huang^{1,*}, Hui Kang^{2,*}, Sheng Liu³, Olivier Salvado¹,
Thierry Rakotoarivelo¹, Dadong Wang¹, Tongliang Liu²

¹ Data61, CSIRO, ² The University of Sydney, ³ NYU Center for Data Science

Abstract

Convolutional Neural Networks (CNNs) have demonstrated superiority in learning patterns, but are sensitive to label noises and may overfit noisy labels during training. The early stopping strategy averts updating CNNs during the early training phase and is widely employed in the presence of noisy labels. Motivated by biological findings that the amplitude spectrum (AS) and phase spectrum (PS) in the frequency domain play different roles in the animal's vision system, we observe that PS, which captures more semantic information, can increase the robustness of DNNs to label noise, more so than AS can. We thus propose early stops at different times for AS and PS by disentangling the features of some layer(s) into AS and PS using Discrete Fourier Transform (DFT) during training. Our proposed Phase-Amplitude Disentangled Early Stopping (PADDLES) method is shown to be effective on both synthetic and real-world label-noise datasets. PADDLES outperforms other early stopping methods and obtains state-of-the-art performance ¹.

1. Introduction

Learning from noisy labels (LNL) [1] is an active area of research within the deep learning community [12, 14, 32, 40, 57, 60, 62]. Noisy labels are common in real-world applications [44, 49, 54, 59], and trustworthy AI should be robust to mislabelling.

It has been argued that CNNs learn first the actual pattern before over fitting the noise [3], which inspired many works in LNL [14, 24, 25, 27, 28, 51, 56]. A training strategy is early stopping (ES), which stops the gradient-based optimization at a specific early training step. Due to its effectiveness, ES is widely applied in current LNL models and has achieved promising performance [4, 24, 27, 33, 46].

The frequency and spatial domains are alternative codes

for depicting signal data such as images and text [36, 45]. Different frequency components contain different information [7]. The amplitude spectrum (AS) quantifies how much of each sinusoidal component is present, while the phase spectrum (PS) reveals the location of each sinusoidal component within an image. Biological justification and psychological patterns testing [13, 42] demonstrate that the response of cells in the primary visual cortex (V1) is closely related to the local AS for specific image patterns (frequency and orientation). That is, the AS component usually represents the intensity of the patterns in the image. On the other hand, previous qualitative and quantitative studies [7, 13] indicate that the PS is the key to locating salient object areas and holds visible structured information for vision recognition [11, 23, 35], thus contains more semantic information than the AS.

As a robust vision system, human vision focuses on semantic parts during object recognition, and relies more on the image components related to the PS than the AS [8, 13, 23, 35]. This system builds a strong connection between semantic feature space and label space, helping humans ‘understand’ the actual correlation between objects and their corresponding identifiers (labels). The human visual system is very robust to label noise. However, CNNs profit from human unperceivable high-frequency information in images [18, 50]. Without adequate regulations, CNNs model the correlation of objects and their labels mainly based on the connection between AS and the given annotations. Such over-dependence is demonstrated as the leading cause of their sensitivity to image perturbation and overconfidence in out-of-distribution (OOD) detection [8]. We argue that CNNs’ over-dependence of connection between the less semantic AS and labels may spoil their recognition robustness, resulting in their vulnerability to label noise.

To investigate the impact of label noise on deep models trained with different image components, we generate symmetric label noise [14, 48] with a 50% noise rate and feed it with raw images, PS and AS to a ResNet-18 [15] model separately. As shown in Figures 1a and 1b, the convergence speed of CNNs on AS and PS differs. When CNNs start

*Co-first authors.

¹Codes will be available upon acceptance.

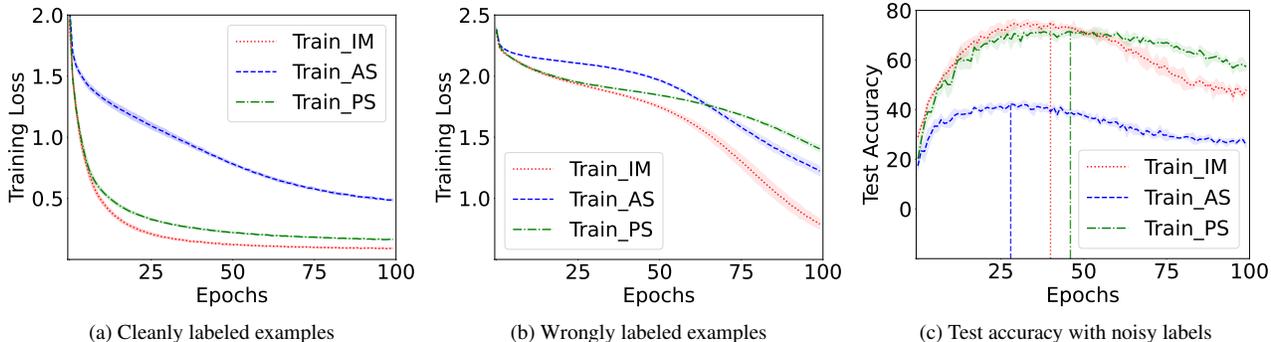


Figure 1. Results of training a ResNet-18 model on CIFAR-10 using original images, amplitude spectrum, and phase spectrum (“Train_IM”, “Train_AS”, and “Train_PS” in the Figure) on cleanly and noisily labeled subsets. The curves are averaged across five random runs. The dotted vertical lines indicate the best performance steps of different image components. The converging speed of the deep model trained on AS and PS differs, especially on wrongly labeled examples. Approaching the end of the training, when the wrong labels begin to be memorized, the model accelerates fitting to AS, resulting in an intersection on the training curves of AS and PS, shown in Figure 1b. Hence, PS can help the deep model become more resistant to label noises than AS.

to overfit the noisy labels, they fit AS much faster than PS (Figure 1b). Meanwhile, the convergence speed on PS is slower than AS and the raw images, which indicates that PS can help the CNNs become more robust towards mislabels than AS or raw inputs. Note that the model trained with only AS or PS performs worse than the one trained with the raw images (Figure 1c). This is not surprising as either AS or PS could miss some information from the original image data. Therefore, an intuitive solution to improve the robustness of the CNNs to the noisy labels is choosing different early stop points for AS and PS, during the training of the CNNs. In this way, we can suppress the over-dependence of CNNs on AS while shift to utilize more PS components.

Current CNNs are trained based on gradients update via backward propagation. The raw images are fixed and do not need gradient computing during the optimization. Therefore, it is hard to control the model optimization on raw AS and PS directly. To tackle this challenge, we propose to use deep features to represent the ‘image’, as each ‘pixel’ of the feature map corresponds to an original image patch. Moreover, a similar study to that shown in Figure 1 for the deep features of ResNet blocks supports our solution. We observe that different frequency components from the deep features hold a similar property to those from the raw image (Please refer to the *supplemental materials* for this study). Specifically, we propose to disentangle the deep image features into AS and PS at different training steps by Discrete Fourier Transform (DFT). We first detach the AS component from the gradient computational graph to stop its involvement in the model update, which can alleviate the potential negative effects of AS in the later training stage. With AS being detached, we continue train the deep model with PS components. The optimization on the PS components will be stopped after a few training epochs. Notice

that the detached components will regenerate the deep features in the spatial domain through inverse DFT (iDFT). This is efficient as there is no modification to the original architecture. Moreover, complete information is used for training. We call the proposed method as Phase-Amplitude DisentangLed Early Stopping (PADDLES). To the best of our knowledge, PADDLES is the first method to consider features learned with noisy labels in the frequency domain and thus is orthogonal to existing methods that mainly focus on the spatial domain. Our contributions are as follows:

- We study learning with noise labels from the frequency domain and find that PS can help CNNs become more resistant to label noise than AS.
- We propose to early stop training at different stages for AS and PS. We demonstrate that our proposed method can benefit from the robustness of the PS without losing information on AS during the training of CNNs.
- Extensive experiments on benchmark datasets such as CIFAR-10/100, CIFAR-10N/100N, and Clothing-1M validate the effectiveness of the proposed method.

2. Related Work

2.1. Learning with noisy labels

Current methods [9, 10, 12, 14, 16, 19, 21, 24, 25, 29–32, 37, 38, 40, 41, 47, 53, 55, 57, 61–65, 67, 68] of learning with noisy labels (LNL) can be grouped into two categories: model-based and model-free approaches.

Model-based methods [29, 37, 56, 57, 63] propose to describe the relations between noisy and clean labels based on the assumption that the noisy label is sampled from a conditional probability distribution on the true labels.

Hence, the core idea of these methods is to estimate the underlying noise transition probabilities. For instance, [12] used a noise adaptation layer on the top of a classification model to learn the transition probabilities. T-revision [58] added fine-tuned slack variables to estimate the noise transition matrix without anchor points. Moreover, a recent work [29] proposed to model the label noise via a sparse over-parameterized term and use implicit algorithmic regularizations to recover the underlying mislabels. These methods hold some assumptions about the noisy label distribution, which may be inapplicable in some scenarios. Our method does not focus on particular label distribution and therefore does not belong to model-based methods.

Instead of modeling the noisy labels directly, model-free methods [4, 14, 24, 56] aim to utilize the memorization effect of deep models to suppress the negative impact of the noisy labels. A representative method is Co-teaching [14], which uses two deep networks to train each other with small-loss instances in mini-batches. DivideMix [24] further extended Co-teaching with two Beta Mixture Models. Moreover, DivideMix imported MixMatch [5] training to utilize the unlabeled (unconfident) samples to boost the deep models. PES [4] investigated the progressive early stopping of deep networks, which selects different early stopping for different parts of the deep model and achieved significant improvement over previous early stopping methods. Unlike existing model-free methods, our method is the first work designed from the data domain’s perspective in frequency representation. Inspired by the biological analysis of the vision system on different spectrums, we find that PS can help CNNs become more resistant to noisy labels than the AS. Therefore, we propose to disentangle the different components of the frequency domain and choose different early stopping strategies, which further exploit the memorization effect and can achieve good performance.

2.2. CNNs with frequency domain

To explain the behavior of CNNs, recent studies provide new insights from the viewpoint of the frequency domain [8, 18, 26, 50]. [50] points out that high-frequency components from an image play significant roles in improving the performance of CNNs. Moreover, [26] investigated the PS in face forgery detection and found that CNNs trained with PS can boost the detection accuracy. APR [8] presented qualitative and quantitative analyses of AS and PS for CNNs and proposed to recombine the AS and PS as a data augmentation method to improve the robustness of the CNNs models to adversarial attack. Inspired by these breakthroughs, we are the first to investigate the frequency domain in learning with noisy labels and find that PS and AS behave differently in the training of CNNs models with mislabels. Furthermore, we propose to dynamically stop training CNN on different frequency components, giving a

new solution to the over-fitting problem of noisy labels.

3. Methodology

3.1. Problem Definition

In learning with noisy labels, the real training data distribution can be defined as $\mathcal{D} = \{(x, y) | x \in \mathcal{X}, y \in \{1, \dots, K\}\}$, where \mathcal{X} is the sample space, and $\{1, \dots, K\}$ denotes the label space with K classes. However, the actual distribution of the label space is usually inaccessible since the data collection and dataset construction will inevitably import label errors. We can only use the accessible noisy dataset $\widehat{\mathcal{D}} = \{(x, \widehat{y}) | x \in \mathcal{X}, \widehat{y} \in \{1, \dots, K\}\}$ to train the model, where \widehat{y} denotes the corrupted labels. The goal of our algorithm is to learn a robust deep classifier from the noisy data that can perform accurately on the query samples.

3.2. Phase-Amplitude Disentangled Early Stopping

Training a deep model with a noisy dataset $\widehat{\mathcal{D}}$ is challenging as the model will fit the clean labels first and then overfit the noisy labels, as shown in Figure 1. This memorization effect motivates previous methods to adapt the early stopping to cease the optimization of deep models at a specific step. Namely, the early stopping method aims to choose a suitable step tp in training a deep model f_{Θ} . The training process is to learn an optimal Θ^* :

$$\Theta^* = \arg \min_{\Theta} \frac{1}{N} \sum_{i=1}^N \mathcal{L}(\widehat{y}_i, f_{\Theta_T} \circ f_{\Theta_{T-1}} \circ \dots \circ f_{\Theta_0}(x_i)), \quad (1)$$

where $\Theta = \{\Theta_T, \Theta_{T-1}, \dots, \Theta_0\}$ denotes the parameters of the deep model, and \circ denotes the operator of the function composition. The deep model $f_{\Theta}(\cdot)$ is rewritten as $f_{\Theta_T} \circ f_{\Theta_{T-1}} \circ \dots \circ f_{\Theta_0}(\cdot)$ since the deep neural networks can be viewed as a stack of non-linear functions. Θ_T denotes the parameter of the T th non-linear function. x_i, \widehat{y}_i represent the i th sample and its label, and \mathcal{L} denotes the training loss.

To obtain Θ^* , previous works [4, 28, 56] developed various optimization policies from the perspective of robust loss function design [28], gradient regulation [56], and progressive architecture selection [4]. These methods focus on the spatial domain, and treat the input data (images) as a whole. However, as discussed in Section 1, different image components play different roles in the vision system. It is undesirable to stop the model optimization on these components simultaneously.

For this reason, we propose to investigate the early stopping on the input data components and select different stop points for different components. It is natural to consider the frequency domain due to its equivalent representation of input data on the spatial domain [7, 36] and the vision

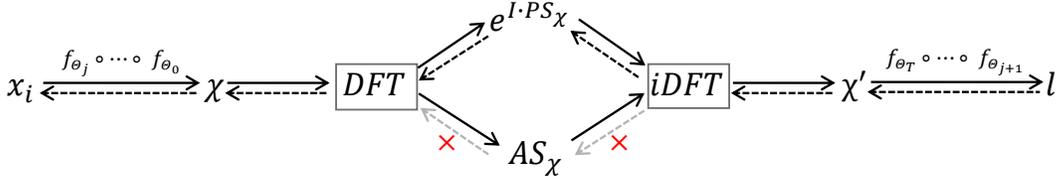


Figure 2. An illustration of the proposed PADDLES strategy stops the amplitude spectrum’s involvement in model training. “ \longrightarrow ” denotes the forward propagation, while the “ \longleftarrow ” represents the backward propagation. Using Equations 2 3 and 4, we form a computational chain to disentangle the frequency domain representation, and then we can stop the backward propagation of the target component. In this way, we can control the model’s optimization with each component and choose different stopping points.

properties of amplitude and phase spectra [6, 23], as discussed previously. Specifically, for an input sample x_i , the deep feature after j th operation in f_{Θ} can be represented as $\chi = f_{\theta_j} \circ \dots \circ f_{\theta_0}(x_i)$, and its frequency domain representation \mathcal{F}_{χ} can be computed using DFT:

$$\mathcal{F}_{\chi}(u) = \sum_{p=0}^{M-1} \chi_p e^{-\frac{I \cdot 2\pi}{M} pu}, \quad (2)$$

which can be denoted as $\mathcal{F}_{\chi} = DFT(\chi)$. u represents a specific frequency, M is the number of sampled points, I is the imaginary unit, and χ_p denotes the value at the position p of χ . We consider one dimension here for simplicity, and the higher-dimensional DFT corresponds to successive Fourier transforms along each dimension in sequence. Notice that the $\mathcal{F}_{\chi}(u)$ is a complex-valued variable, its real part can be denoted as $Real_{\mathcal{F}_{\chi}}$, and the imaginary part is $Imag_{\mathcal{F}_{\chi}}$. We then disentangle the phase and amplitude components using the following rules:

$$\begin{aligned} \mathcal{P}S_{\chi}(u) &= \arctan\left(\frac{Imag_{\mathcal{F}_{\chi}}(u)}{Real_{\mathcal{F}_{\chi}}(u)}\right), \\ \mathcal{A}S_{\chi}(u) &= |\mathcal{F}_{\chi}(u)|, \end{aligned} \quad (3)$$

where $\mathcal{P}S_{\chi}$ represents the phase spectrum, $\mathcal{A}S_{\chi}$ represents the amplitude spectrum, $\arctan(\cdot)$ is the inverse trigonometric function, and $|\cdot|$ computes the absolute value. Using Equations 2 and 3, the deep features are decomposed into amplitude and phase components during the model training. Afterward, we restore the deep feature using iDFT:

$$\chi'_p = \frac{1}{M} \sum_{u=0}^{M-1} (e^{I \cdot \mathcal{P}S_{\chi}(u)} \odot \mathcal{A}S_{\chi}(u)) e^{\frac{I \cdot 2\pi}{M} pu}, \quad (4)$$

which can be represented with $\chi' = iDFT(e^{I \cdot \mathcal{P}S_{\chi}} \odot \mathcal{A}S_{\chi})$. Notice that $\chi' = \chi$, \odot indicates the element-wise multiplication operation.

Through Equation 2, 3, and 4, we construct a computation flow disentangling the phase spectrum $\mathcal{P}S_{\chi}$ and the

Algorithm 1: Model Update with AS/PS control

Input : A noisy set \hat{D} , Disentangle point j , Deep Model $f_{\Theta} = \{\theta_T, \theta_{T-1}, \dots, \theta_0\}$, Target Spectrum ($\mathcal{A}S_{\chi}$ or $\mathcal{P}S_{\chi}$).

- 1 Extract χ at f_{θ_j} , disentangle χ into $\mathcal{A}S_{\chi}$ and $\mathcal{P}S_{\chi}$ using Equation 2 and 3;
- 2 Detach gradient computation of the target Spectrum ($\mathcal{A}S_{\chi}$ or $\mathcal{P}S_{\chi}$) in Equation 3;
- 3 Restore deep feature χ' using Equation 4;
- 4 Update network parameter Θ using Equation 1;

Output: The updated model $f_{\Theta'}$.

amplitude spectrum $\mathcal{A}S_{\chi}$ from the original feature χ during the end-to-end model training. Therefore, we can control the deep model’s optimization with each component. Specifically, the end-to-end training of a deep model f_{Θ} consists of the forward and the backward propagations, the forward propagation (right arrows in Figure 2) will generate the intermediate values ($\chi, \mathcal{P}S_{\chi}, \mathcal{A}S_{\chi}, \chi'$) with the input x_i , and the backward propagation (left arrows in Figure 2) will track the gradients for each intermediate value and model parameter. Finally, the model is updated using the gradient descent with the tracked gradients. For the backward propagation of f_{Θ} , we need to compute the partial derivatives of loss function \mathcal{L} with respect to $\mathcal{P}S_{\chi}$ ($\frac{\partial \mathcal{L}}{\partial \mathcal{P}S_{\chi}}$) and $\mathcal{A}S_{\chi}$ ($\frac{\partial \mathcal{L}}{\partial \mathcal{A}S_{\chi}}$)². Stopping computing these derivatives can detach the phase-related gradient or amplitude-related gradient nodes from the gradient computational graph and thus control the model optimization on each frequency component, as illustrated in Figure 2.

3.3. Practical Implementation

The proposed PADDLES is illustrated in Algorithm 1 and Algorithm 2. In this section, we introduce the structure of

²Thanks to the automatic differentiation engine of deep learning frameworks, e.g., PyTorch and TensorFlow, it is convenient to obtain the derivatives and gradient for each variable. Therefore, we omit the derivatives computation of PS and AS here.

Algorithm 2: PADDLES

Input : A noisy set $\hat{\mathcal{D}}$, Algorithm 1, Model $f_{\Theta=\{\Theta_T, \Theta_{T-1}, \dots, \Theta_0\}}$, Disentangle point j , \mathcal{AS}_χ training epoch T_A , \mathcal{PS}_χ training epoch T_P , Additional epoch T_0 , Epochs for remaining part: T_{j+1}, \dots, T_T .

- 1 **for** $i = 1$ **to** T_A **do**
- 2 | Update model parameter Θ using Equation 1;
- 3 **end**
- 4 **for** $i = 1$ **to** T_P **do**
- 5 | Update model parameter using Algorithm 1 with \mathcal{AS}_χ detached;
- 6 **end**
- 7 **for** $i = 1$ **to** T_0 **do**
- 8 | Update model parameter using Algorithm 1 with \mathcal{PS}_χ detached;
- 9 **end**
- 10 Hook \mathcal{AS}_χ and \mathcal{PS}_χ to the gradient computation graph during backpropagation;
- 11 **for** $l = j + 1$ **to** T **do**
- 12 | Freeze $\{\Theta_0, \dots, \Theta_j\}$ and re-initialize other parameters;
- 13 | **for** $i = 1$ **to** T_l **do**
- 14 | | Update network parameter $\{\Theta_{j+1}, \dots, \Theta_T\}$ using Equation 5;
- 15 | **end**
- 16 **end**

Output: The optimized model f_{Θ^*} .

our model and the corresponding learning settings.

To reduce the difficulty of implementation and further improve the robustness of PADDLES, we incorporate progressive early stopping (PES) [4] in our model training. Therefore, we need to add a copy of the PES optimization strategy.

After finishing the amplitude and phase spectrum training (Step 9 in Algorithm 2). The parameter parts $\{\Theta_0^*, \dots, \Theta_j^*\}$ are well-optimized. We then apply PES to update the remaining parts $\{\Theta_{j+1}, \dots, \Theta_T\}$ with previous parameters fixed. T_l steps will be performed during training using the following objective:

$$\min_{\{\Theta_{l, \dots, T}\}} \frac{1}{N} \sum_{i=1}^N \mathcal{L} \left(\hat{y}_i, f_{\Theta_T} \circ \dots \circ f_{\Theta_l} \circ f_{\Theta_{l-1}^*} \circ \dots \circ f_{\Theta_0^*} (x_i) \right),$$
$$l = j + 1, j + 2, \dots, T - 1, T. \quad (5)$$

After the optimization with Equation 5, the final model $f_{\Theta^*=\{\Theta_0^*, \dots, \Theta_T^*\}}$ is obtained.

Learning Settings We adopt PADDLES as a confident sample selector to boost noisy label learning with super-

vised and semi-supervised learning settings. The confident sample set \mathcal{D}_{lb} is defined as

$$\mathcal{D}_{lb} = \{(x_i, \hat{y}_i) | \hat{y}_i = \bar{y}_i, i = 1, \dots, N\},$$
$$\bar{y}_i = \arg \max_{\tau \in \{1, \dots, K\}} \frac{1}{2} [f_{\Theta^*}^\tau(A(x_i)) + f_{\Theta^*}^\tau(A'(x_i))], \quad (6)$$

where A and A' are data augmentation operators randomly sampled from the same augmentation set, $f_{\Theta^*}^\tau(x_i)$ indicates the classification probability of x_i belonging to class τ . For the supervised learning with confident samples, we adopt the weighted classification loss in the training.

For the semi-supervised setting, besides the confident label set \mathcal{D}_{lb} , the additional unlabeled set \mathcal{D}_{ub} is defined as

$$\mathcal{D}_{ub} = \{x_i | \hat{y}_i \neq \bar{y}_i, i = 1, \dots, N\},$$
$$\bar{y}_i = \arg \max_{\tau \in \{1, \dots, K\}} \frac{1}{2} [f_{\Theta^*}^\tau(A(x_i)) + f_{\Theta^*}^\tau(A'(x_i))]. \quad (7)$$

We adopt the MixMatch [5] loss in the semi-supervised learning as previous works [4, 24].

4. Experiments

4.1. Experimental Setup

Datasets: We demonstrate the effectiveness of our PADDLES on the two manually corrupted datasets: CIFAR-10 and CIFAR-100 [22], and two real-world noisy sets: CIFAR-N [53] and Clothing-1M [59]. CIFAR-10 and CIFAR-100 contain 50k training samples and 10k testing samples. CIFAR-10 has 10 classes, while CIFAR-100 contains 100 classes. The original labels of these two datasets are clean. We generate three types of noisy labels, *i.e.*, symmetric, pairflip, and instance-dependent label noise, according to [14, 28, 56, 58]. CIFAR-N consists of CIFAR-10N and CIFAR-100N, datasets of re-annotated CIFAR-10 and CIFAR-100 by human annotators. Specifically, CIFAR-10N has five types of labels: *Random 1*, *Random 2*, *Random 3*, *Aggregate*, and *Worst*, which are derived from three submitted label sets. CIFAR-100N contains a single human annotated label set named *Noisy Fine*. Clothing-1M has one million clothing images in 14 classes clawed from online shopping web sites. The labels of Clothing-1M are generated according to the context on the shopping web pages, resulting in lots of mislabelled samples. This dataset also provides 14,313 and 10,526 images with clean labels for validation and testing. We apply the random crop and random horizontal flip as data augmentations for learning with confident samples, and add MixUp [66] data augmentation for semi-supervised settings. For CIFAR-N dataset, we use a CIFAR-10 augmentation policy from [34]. The input image size of CIFAR-like datasets is set to 32×32 . For the Clothing-1M dataset, we first resize input images to the size

Table 1. Comparison with different methods under supervised learning of confident samples on CIFAR. The results of the baseline methods are taken from [4]. The best results are in bold. Mean and standard deviation computed over five independent runs are reported.

Dataset	Method	Symmetric		Pairflip	Instance	
		20%	50%	45%	20%	40%
CIFAR-10	CE	84.00±0.66	75.51±1.24	63.34±6.03	85.10±0.68	77.00±2.17
	Co-teaching	87.16±0.11	72.80±0.45	70.11±1.16	86.54±0.11	80.98±0.39
	Forward-T	85.63±0.52	77.92±0.66	60.15±1.97	85.29±0.38	74.72±3.24
	JointOptim	89.70±0.11	85.00±0.17	82.63±1.38	89.69±0.42	82.62±0.57
	T-revision	89.63±0.13	83.40±0.65	77.06±6.47	90.46±0.13	85.37±3.36
	DMI	88.18±0.36	78.28±0.48	57.60±14.56	89.14±0.36	84.78±1.97
	CDR	89.72±0.38	82.64±0.89	73.67±0.54	90.41±0.34	83.07±1.33
	PES	92.38±0.40	87.45±0.35	88.43±1.08	92.69±0.44	89.73±0.51
	PADDLES	92.43±0.18	87.94±0.22	89.32±0.21	92.76±0.30	89.87±0.51
CIFAR-100	CE	51.43±0.58	37.69±3.45	34.10±2.04	52.19±1.42	42.26±1.29
	Co-teaching	59.28±0.47	41.37±0.08	33.22±0.48	57.24±0.69	45.69±0.99
	Forward-T	57.75±0.37	44.66±1.01	27.88±0.80	58.76±0.66	44.50±0.72
	JointOptim	64.55±0.38	50.22±0.41	42.61±0.61	65.15±0.31	55.57±0.41
	T-revision	65.40±1.07	50.24±1.45	41.10±1.95	60.71±0.73	51.54±0.91
	DMI	58.73±0.70	44.25±1.14	26.90±0.45	58.05±0.20	47.36±0.68
	CDR	66.52±0.24	55.30±0.96	43.87±1.35	67.33±0.67	55.94±0.56
	PES	68.89±0.45	58.90±2.72	57.18±1.44	70.49±0.79	65.68±1.41
	PADDLES	69.19±0.88	59.78±3.15	58.68±1.28	70.88±0.55	66.11±1.19

Table 2. Comparison with different methods under semi-supervised learning of confident samples on CIFAR. The results of the baseline methods are taken from [4]. The best results are in bold. Mean and standard deviation computed over five independent runs are reported.

Dataset	Method	Symmetric			Pairflip	Instance	
		20%	50%	80%	45%	20%	40%
CIFAR-10	CE	86.5±0.6	80.6±0.2	63.7±0.8	74.9±1.7	87.5±0.5	78.9±0.7
	MixUp	93.2±0.3	88.2±0.3	73.3±0.3	82.4±1.0	93.3±0.2	87.6±0.5
	DivideMix	95.6±0.1	94.6±0.1	92.9±0.3	85.6±1.7	95.5±0.1	94.5±0.2
	ELR+	94.9±0.2	93.6±0.1	90.4±0.2	86.1±1.2	94.9±0.1	94.3±0.2
	PES	95.9±0.1	95.1±0.2	93.1±0.2	94.5±0.3	95.9±0.1	95.3±0.1
	PADDLES	96.1±0.1	95.3±0.2	93.3±0.1	94.6±0.1	96.2±0.1	95.5±0.2
CIFAR-100	CE	57.9±0.4	47.3±0.2	22.3±1.2	38.5±0.6	56.8±0.4	48.2±0.5
	MixUp	69.5±0.2	57.1±0.6	34.1±0.6	44.2±0.5	67.1±0.1	55.0±0.1
	DivideMix	75.3±0.1	72.7±0.6	56.4±0.3	48.2±1.0	75.2±0.2	70.9±0.1
	ELR+	75.5±0.2	71.0±0.2	50.4±0.8	65.3±1.3	75.8±0.1	74.3±0.3
	PES	77.4±0.3	74.3±0.6	61.6±0.6	73.6±1.7	77.6±0.3	76.1±0.4
	PADDLES	77.9±0.1	74.8±0.3	62.9±0.3	74.7±1.5	77.7±0.3	76.3±0.1

of 256×256 , then randomly crop the image to 224×224 , and horizontally flip the images with a random probability.

Comparison Methods: We compared the proposed PADDLES with the following approaches: 1) Cross Entropy (CE) and MixUp as two baselines, with which the deep models were trained with cross-entropy loss and mixup [66] strategy, respectively. 2) Classic LNL methods: Co-teaching [14], Forward-T [37], JointOptim [46], T-revision [58], M-correction [2], DMI [60] and JoCoR [52]. 3) State-of-the art LNL methods: DivideMix [24], CDR [56], ELR [28], PES [4], CORES [9] and SOP [29].

Model Structures and Hyperparameters: We implemented our method with PyTorch. The compared methods were implemented or re-implemented based on open-source codes and original papers with same hyperparameters.

For the supervised learning, we use ResNet-18 and

ResNet-34 architectures for CIFAR-10 and CIFAR-100, respectively. The disentangle point j is between the 3rd and 4th ResNet blocks. We train the networks 110 epochs with the following parameters: the initial learning rate is 0.1, a weight decay of 10^{-4} , and a batch size of 128. For PES training policy, we use the default parameters in the paper. Different types and levels of label noises result in different converge points of the CNNs on AS and PS. Therefore, we set different stopping points of T_A and T_P for the different label noises. For CIFAR-10, the T_A for 20%/40% Instance noise, 45% Pairflip noise, and 20%/50% Symmetric noise are 17, 20, 19, 18, 19, respectively. The corresponding T_P are 13, 25, 16, 21, 20. For CIFAR-100, the T_A for 20%/40% Instance noise, 45% Pairflip noise, and 20%/50% Symmetric noise are 20, 20, 19, 29, 20, respectively. The corresponding T_P are 22, 22, 26, 11, 13. The T_0 in Algorithm 2

Table 3. Comparison with state-of-the-art methods on CIFAR-N. Mean and standard deviation over five runs are reported. The results of the baseline methods are taken from the leaderboard in [53]. We use ResNet-34 as backbone like other methods expect for SOP+, which adopted PreActResNet-18.

Method	CIFAR-10N					CIFAR-100N
	Random 1	Random 2	Random 3	Aggregate	Worst	Noisy Fine
CE	85.02±0.65	86.46±1.79	85.16±0.61	87.77±0.38	77.69±1.55	55.50±0.66
Forward-T	86.88±0.50	86.14±0.24	87.04±0.35	88.24±0.22	79.79±0.46	57.01±1.03
T-revision	88.33±0.32	87.71±1.02	87.79±0.67	88.52±0.17	80.48±1.20	51.55±0.31
Co-Teaching	90.33±0.13	90.30±0.17	90.15±0.18	91.20±0.13	83.83±0.13	60.37±0.27
ELR+	94.43±0.41	94.20±0.24	94.34±0.22	94.83±0.10	91.09±1.60	66.72±0.07
CORES*	94.45±0.14	94.88±0.31	94.74±0.03	95.25±0.09	91.66±0.09	55.72±0.42
DivideMix	95.16±0.19	95.23±0.07	95.21±0.14	95.01±0.71	92.56±0.42	71.13±0.48
PES	95.06±0.15	95.19±0.23	95.22±0.13	94.66±0.18	92.68±0.22	70.36±0.33
SOP+	95.28±0.13	95.31±0.10	95.39±0.11	95.61±0.13	93.24±0.21	67.81±0.23
PADDLES	95.86±0.12	96.03±0.16	95.97±0.15	95.46±0.14	93.85±0.34	71.32±0.36

Table 4. Comparison with different methods of test accuracy on Clothing-1M. All methods use a pretrained ResNet-50 architecture. Results of other methods are taken from the original papers. * indicates that the methods are based on an ensemble model, while other methods are obtained with a single network.

CE	Forward-T	JoCoR	JointOptim	DMI
69.21	69.84	70.30	72.16	72.46
ELR	CORES ²	SOP	T-revision	PES
72.87	73.24	73.50	74.18	74.64
DivideMix*	ELR+*	PES*	PADDLES	PADDLES*
74.76	74.81	74.99	74.90	75.07

is set to 0.

For the semi-supervised learning, we use PreAct ResNet-18 for CIFAR-10 and CIFAR-100, and use ResNet-34 for CIFAR-N. For Clothing-1M, we adopt the ResNet-50 pretrained on the ImageNet. The disentangle point j is set between the 3rd and 4th ResNet blocks. We train the model 500/300 epochs using cosine annealing strategy for CIFAR/CIFAR-N datasets, and the initial learning rate is 0.02, with a weight decay of 5×10^{-4} , stopping points of \mathcal{AS}_χ , \mathcal{PS}_χ are set to 30 ($T_A = 30$) and 35 ($T_P = 5$), respectively. T_0 is set to 1, and we do observe further performance improvement with a larger T_0 like 5 in our CIFAR-N settings. For Clothing-1M, we train the model with 150 epochs and use a three phase OneCycle [43] scheduler to dynamically adjust the learning rate with the max learning rate of 8.55×10^{-3} . We set the learning rate to 4.5×10^{-3} with a weight decay of 0.001, stopping points of \mathcal{AS}_χ , \mathcal{PS}_χ are set to 10 ($T_A = 10$) and 29 ($T_P = 19$), respectively. More details can be found in the *Supplemental Materials*.

4.2. Classification Performance on Noisy Datasets

Results on Synthetic Datasets: We evaluate PADDLES on CIFAR-10 and CIFAR-100 with different levels and types of label noise under supervised learning, as shown in Table 1. Under the same architectures, PADDLES out-

performs the other methods across different noisy types and noisy levels, which demonstrates its effectiveness.

In Table 2, we compare PADDLES with state-of-the-art semi-supervised LNL methods. PADDLES achieves a significant performance improvement of around 10% to 40% over the baseline methods such as CE and MixUp. Moreover, PADDLES beats the state-of-the-art LNL methods like ELR+ and PES on all settings. Specifically, with 80% Symmetric label noise on CIFAR-100, the classification accuracies are 62.9% vs. 61.6% PES, indicating the superiority of PADDLES in using unlabelled data to boost classification performance.

Results on Real-world Datasets: We compare the classification performance of various methods on Clothing-1M in Table 4. All of the compared methods adopt a pre-trained ResNet-50 backbone on the ImageNet. Since PADDLES is equipped with a more nuanced optimization strategy from perspectives of frequency domain and progressive model construction, it achieves state-of-the-art performance.

Furthermore, we test our PADDLES model on a more challenging real-world noise-label dataset, as shown in Table 3. CIFAR-N consists of CIFAR-10N and CIFAR-100N with six types of noisy labels annotated by human observers. We can observe a performance gain of PADDLES by comparing different methods on five types of labels except for CIFAR-10N’ Aggregate. PADDLES achieves comparable performance towards SOP+ on CIFAR-10N’s Aggregate labels.

4.3. Ablation Studies

We analyze different components of the PADDLES and summarize the results in Table 5. It can be observed that without PES on updating the latter parts of the model, PADDLES.Base achieves a significant improvement over the baseline CE method. Compared with other state-of-the-art methods, PADDLES.Base obtains comparable performance. For instance, with 45% Pairflip label noise, PAD-

Table 5. Ablation studies about the proposed PADDLES under the supervised setting, experiments on CIFAR-10 are based on a ResNet-18 backbone, and experiments on CIFAR-100 are based on a ResNet-34 backbone. PADDLE_Base denotes the model without using the PES strategy to train the latter parts of the model $\{f_{\Theta_{j+1}}, \dots, f_{\Theta_T}\}$ in Equation 5.

Dataset	Method	Symmetric	Pairflip	Instance
		50%	45%	40%
CIFAR-10	CE	75.51±1.24	63.34±6.03	77.00±2.17
	PADDLES_Base	83.40±0.78	82.80±2.02	85.20±0.47
	PADDLES	87.94±0.22	89.32±0.21	89.87±0.51
CIFAR-100	CE	37.69±3.45	34.10±2.04	42.26±1.29
	PADDLES_Base	47.72±3.55	42.17±2.15	54.68±1.36
	PADDLES	59.78±3.15	58.68±1.28	66.11±1.19

DLES_Base ranks 3rd and 5th among all ten methods on CIFAR-10 and CIFAR-100, as demonstrated in Table 1. After incorporating PES training in the latter model parts, the PADDLES obtains further improvement and achieves state-of-the-art performance since the proposed training policy is designed from the view of the data frequency domain, which is orthogonal to the PES strategy.

Another important component of the PADDLES is the frequency disentangle position j , as presented in Algorithm 2. We choose ResNet models as the backbone and disentangle the deep features at each ResNet block. For example, ‘P1’ indicates decomposing the features before block 1, ‘P5’ is after block 4, and ‘ALL’ means decomposing the features at all five positions. As shown in Figure 3a, we observe that the performance of PADDLES is more stable on CIFAR-10 than on CIFAR-100 at different positions. The best performances are achieved at P3 and P4.

We investigate the hyper-parameter sensitivity of the early stopping points for amplitude spectrum T_A and phase spectrum T_P in Figure 3b and Figure 3c. All experiments are conducted on CIFAR-N datasets with a ResNet-34 backbone. We vary T_A from 18 to 30 with $T_P = 5$ in Figure 3b and set T_P from 5 to 17 with $T_A = 30$. We observe that with fixed T_P , the performance will generally increase when T_A is growing for both Fine noises on CIFAR-100N and Worst noises on CIFAR-10N. When the T_A is fixed, very large training steps for PS will result in performance degradation, as the model starts to overfit the label noises. Moreover, The performances of our model on CIFAR-10N dataset with Aggregate noise stay comparatively stable compared with other noises. The model achieves the best performance with $T_A = 30$ and $T_P = 5$. We also conduct the study on the quality of the learned confident samples and training time of PADDLES, due to the page limit, we include these experiments in the *Supplemental Materials*.

4.4. PADDLES on Text Data

Besides the image data, we find PADDLES is effective for the text data. And we present a text classification ex-

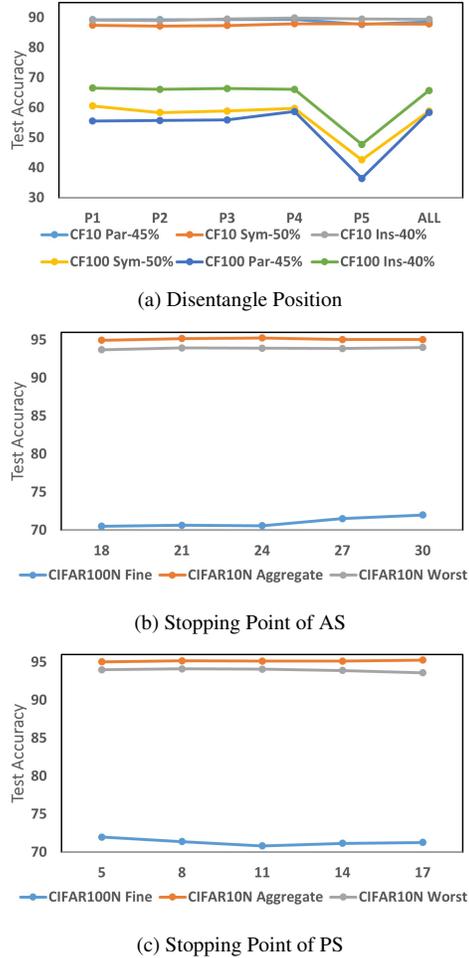


Figure 3. Sensitivity analysis for different choices of disentangle positions, early stopping points of AS, and early stopping points of PS. The Y-axis of each figure represents the testing accuracy (%).

periment using the NEWS [64] dataset in the *Supplemental Materials*. The theoretical and biological foundation of the role of different frequency components in text data is still unclear. Therefore, we leave this for future studies.

5. Conclusion

The performance of deep models is impacted less by label noises if trained on PS than AS, resulting in a different fit speed. Therefore, we propose PADDLES to disentangle the AS and PS from the deep image features and separately detach their backpropagation. This way, PADDLES avoids concurrently stopping the model training of different spectra and thus achieves better performance. Extensive experiments on different types of data (images and texts) with different network architectures (CNNs and MLP) demonstrate the effectiveness of PADDLES which achieves state-of-the-art performance on five noisy label benchmarks.

References

- [1] Dana Angluin and Philip Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988. [1](#)
- [2] Eric Arazo, Diego Ortego, Paul Albert, Noel O’Connor, and Kevin McGuinness. Unsupervised label noise modeling and loss correction. In *ICML*, pages 312–321. PMLR, 2019. [6](#)
- [3] Devansh Arpit, Stanislaw 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 *ICML*, pages 233–242. PMLR, 2017. [1](#)
- [4] Yingbin Bai, Erkun Yang, Bo Han, Yanhua Yang, Jiatong Li, Yinian Mao, Gang Niu, and Tongliang Liu. Understanding and improving early stopping for learning with noisy labels. *NeurIPS*, 34:24392–24403, 2021. [1](#), [3](#), [5](#), [6](#), [11](#), [13](#), [15](#), [16](#)
- [5] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *NeurIPS*, 2019. [3](#), [5](#), [11](#)
- [6] Peng Bian and Liming Zhang. Biological plausibility of spectral domain approach for spatiotemporal visual saliency. In *International conference on neural information processing*, pages 251–258. Springer, 2008. [4](#)
- [7] Kenneth R Castleman. *Digital image processing*. Prentice Hall Press, 1996. [1](#), [3](#)
- [8] Guangyao Chen, Peixi Peng, Li Ma, Jia Li, Lin Du, and Yonghong Tian. Amplitude-phase recombination: Rethinking robustness of convolutional neural networks in frequency domain. In *ICCV*, pages 458–467, 2021. [1](#), [3](#)
- [9] Hao Cheng, Zhaowei Zhu, Xingyu Li, Yifei Gong, Xing Sun, and Yang Liu. Learning with instance-dependent label noise: A sample sieve approach. *ICLR*, 2021. [2](#), [6](#)
- [10] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, and Chiyuan Zhang. Deep learning with label differential privacy. *NeurIPS*, 34:27131–27145, 2021. [2](#)
- [11] Dennis C Ghiglia and Mark D Pritt. Two-dimensional phase unwrapping: theory, algorithms, and software. *A Wiley Interscience Publication*, 1998. [1](#)
- [12] Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. *ICLR*, 2017. [1](#), [2](#), [3](#)
- [13] Chenlei Guo, Qi Ma, and Liming Zhang. Spatio-temporal saliency detection using phase spectrum of quaternion fourier transform. In *CVPR*, pages 1–8. IEEE, 2008. [1](#)
- [14] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *NeurIPS*, 31, 2018. [1](#), [2](#), [3](#), [5](#), [6](#), [11](#), [13](#)
- [15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, June 2016. [1](#), [11](#)
- [16] Wei Hu, Zhiyuan Li, and Dingli Yu. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. In *ICLR*, 2020. [2](#)
- [17] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, pages 646–661. Springer, 2016. [11](#)
- [18] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *NeurIPS*, 32, 2019. [1](#), [3](#)
- [19] 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 *ICML*, pages 2304–2313. PMLR, 2018. [2](#)
- [20] Thorsten Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *ICML*, pages 143–151, 1997. [15](#)
- [21] Jan Kremer, Fei Sha, and Christian Igel. Robust active label correction. In *AISTATS*, pages 308–316. PMLR, 2018. [2](#)
- [22] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009. [5](#)
- [23] Jia Li, Ling-Yu Duan, Xiaowu Chen, Tiejun Huang, and Yonghong Tian. Finding the secret of image saliency in the frequency domain. *IEEE TPAMI*, 37(12):2428–2440, 2015. [1](#), [4](#)
- [24] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *ICLR*, 2020. [1](#), [2](#), [3](#), [5](#), [6](#)
- [25] Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *AISTATS*, pages 4313–4324. PMLR, 2020. [1](#), [2](#)
- [26] Honggu Liu, Xiaodan Li, Wenbo Zhou, Yuefeng Chen, Yuan He, Hui Xue, Weiming Zhang, and Nenghai Yu. Spatial-phase shallow learning: rethinking face forgery detection in frequency domain. In *CVPR*, pages 772–781, 2021. [3](#)
- [27] Sheng Liu, Kangning Liu, Weicheng Zhu, Yiqiu Shen, and Carlos Fernandez-Granda. Adaptive early-learning correction for segmentation from noisy annotations. *CVPR*, 2022. [1](#)
- [28] Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *NeurIPS*, 2020. [1](#), [3](#), [5](#), [6](#)
- [29] Sheng Liu, Zhihui Zhu, Qing Qu, and Chong You. Robust training under label noise by over-parameterization. In *ICML*, 2022. [2](#), [3](#), [6](#)
- [30] Yang Liu and Hongyi Guo. Peer loss functions: Learning from noisy labels without knowing noise rates. In *ICML*, pages 6226–6236. PMLR, 2020. [2](#)
- [31] Yueming Lyu and Ivor W Tsang. Curriculum loss: Robust learning and generalization against label corruption. In *ICLR*, 2020. [2](#)
- [32] Eran Malach and Shai Shalev-Shwartz. “Decoupling” when to update” from” how to update”. *NeurIPS*, 30, 2017. [1](#), [2](#)
- [33] Duc Tam Nguyen, Chaithanya Kumar Mummadi, Thi Phuong Nhung Ngo, Thi Hoai Phuong Nguyen, Laura Beggel, and Thomas Brox. Self: Learning to filter noisy labels with self-ensembling. In *ICLR*, 2020. [1](#)

- [34] Kento Nishi, Yi Ding, Alex Rich, and Tobias Hollerer. Augmentation strategies for learning with noisy labels. In *CVPR*, pages 8022–8031, 2021. [5](#), [11](#)
- [35] Alan V Oppenheim and Jae S Lim. The importance of phase in signals. *Proceedings of the IEEE*, 69(5):529–541, 1981. [1](#)
- [36] Alan V Oppenheim, Alan S Willsky, Syed Hamid Nawab, Gloria Mata Hernández, et al. *Signals & systems*. Pearson Educación, 1997. [1](#), [3](#)
- [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 *CVPR*, pages 1944–1952, 2017. [2](#), [6](#)
- [38] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *NeurIPS*, 2021. [2](#)
- [39] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543, 2014. [15](#)
- [40] Scott E Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *ICLR (Workshop)*, 2015. [1](#), [2](#)
- [41] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *ICML*, pages 4334–4343. PMLR, 2018. [2](#)
- [42] Eero P Simoncelli and Odelia Schwartz. Modeling surround suppression in v1 neurons with a statistically derived normalization model. *NeurIPS*, pages 153–159, 1999. [1](#)
- [43] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE, 2019. [7](#), [12](#)
- [44] Zeren Sun, Yazhou Yao, Xiu-Shen Wei, Yongshun Zhang, Fumin Shen, Jianxin Wu, Jian Zhang, and Heng Tao Shen. Webly supervised fine-grained recognition: Benchmark datasets and an approach. In *ICCV*, pages 10602–10611, 2021. [1](#)
- [45] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010. [1](#)
- [46] Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *CVPR*, pages 5552–5560, 2018. [1](#), [6](#)
- [47] Kiran K Thekumparampil, Ashish Khetan, Zinan Lin, and Sewoong Oh. Robustness of conditional gans to noisy labels. *NeurIPS*, 31, 2018. [2](#)
- [48] Brendan Van Rooyen, Aditya Menon, and Robert C Williamson. Learning with symmetric label noise: The importance of being unhinged. *NeurIPS*, 28, 2015. [1](#), [11](#)
- [49] Sudheendra Vijayanarasimhan and Kristen Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. *IJCV*, 108(1):97–114, 2014. [1](#)
- [50] Haohan Wang, Xindi Wu, Zeyi Huang, and Eric P Xing. High-frequency component helps explain the generalization of convolutional neural networks. In *ICCV*, pages 8684–8694, 2020. [1](#), [3](#)
- [51] Yisen Wang, Weiyang Liu, Xingjun Ma, James Bailey, Hongyuan Zha, Le Song, and Shu-Tao Xia. Iterative learning with open-set noisy labels. In *CVPR*, pages 8688–8696, 2018. [1](#)
- [52] Hongxin Wei, Lei Feng, Xiangyu Chen, and Bo An. Combating noisy labels by agreement: A joint training method with co-regularization. In *CVPR*, 2020. [6](#)
- [53] Jiaheng Wei, Zhaowei Zhu, Hao Cheng, Tongliang Liu, Gang Niu, and Yang Liu. Learning with noisy labels revisited: A study using real-world human annotations. In *ICLR*, 2022. [2](#), [5](#), [7](#)
- [54] Peter Welinder and Pietro Perona. Online crowdsourcing: rating annotators and obtaining cost-effective labels. In *CVPR Workshops*, pages 25–32. IEEE, 2010. [1](#)
- [55] Songhua Wu, Mingming Gong, Bo Han, Yang Liu, and Tongliang Liu. Fair classification with instance-dependent label noise. In *Conference on Causal Learning and Reasoning*, pages 927–943. PMLR, 2022. [2](#)
- [56] Xiaobo Xia, Tongliang Liu, Bo Han, Chen Gong, Nannan Wang, Zongyuan Ge, and Yi Chang. Robust early-learning: Hindering the memorization of noisy labels. In *ICLR*, 2020. [1](#), [2](#), [3](#), [5](#), [6](#)
- [57] 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. *NeurIPS*, 33:7597–7610, 2020. [1](#), [2](#), [11](#), [13](#)
- [58] Xiaobo Xia, Tongliang Liu, Nannan Wang, Bo Han, Chen Gong, Gang Niu, and Masashi Sugiyama. Are anchor points really indispensable in label-noise learning? *NeurIPS*, 32, 2019. [3](#), [5](#), [6](#)
- [59] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *CVPR*, 2015. [1](#), [5](#)
- [60] Yilun Xu, Peng Cao, Yuqing Kong, and Yizhou Wang. L_{dm}i: A novel information-theoretic loss function for training deep nets robust to label noise. *NeurIPS*, 2019. [1](#), [6](#)
- [61] Erkun Yang, Dongren Yao, Tongliang Liu, and Cheng Deng. Mutual quantization for cross-modal search with noisy labels. In *CVPR*, pages 7551–7560, 2022. [2](#)
- [62] Yu Yao, Tongliang Liu, Mingming Gong, Bo Han, Gang Niu, and Kun Zhang. Instance-dependent label-noise learning under a structural causal model. *NeurIPS*, 2021. [1](#), [2](#)
- [63] Yu Yao, Tongliang Liu, Bo Han, Mingming Gong, Jiankang Deng, Gang Niu, and Masashi Sugiyama. Dual t: Reducing estimation error for transition matrix in label-noise learning. *NeurIPS*, 33:7260–7271, 2020. [2](#)
- [64] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *ICML*, pages 7164–7173. PMLR, 2019. [2](#), [8](#), [15](#), [16](#)
- [65] Xiyu Yu, Tongliang Liu, Mingming Gong, and Dacheng Tao. Learning with biased complementary labels. In *ECCV*, pages 68–83, 2018. [2](#)
- [66] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. [5](#), [6](#), [11](#)

- [67] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *NeurIPS*, 31, 2018. 2
- [68] Zhaowei Zhu, Tongliang Liu, and Yang Liu. A second-order approach to learning with instance-dependent label noise. In *CVPR*, pages 10113–10123, 2021. 2

6. Supplemental Materials

6.1. Study of Deep Features on the Frequency Domain

To investigate the impact of label noise on CNNs trained with different frequency components from different CNNs layers, we conducted several experiments by training a ResNet-18 model [15] with different label noises. We generate three label noises: 50% Symmetric noise [14,48], 40% Instance noise [57], and 45% Pairflip noise [14]. Afterward, we trained the ResNet-18 model under these label noises by adopting PADDLES (Algorithm 1 in our paper) to disentangle and detach the AS/PS components of the deep features from different ResNet-18 blocks during the CNNs training. As the ResNet-18 has four ResNet blocks, we present all deep features extracted from those blocks under three label noises in Figure 4 (50% Symmetric label noise), Figure 5 (40% Instance label noise), and Figure 6 (45% Pairflip label noise). As shown in these Figures, the deep features extracted by different ResNet-18 blocks share a similar behavior with original images that PS components of deep features can help the CNNs become more robust towards label noises than AS or raw deep features. These results strongly support the rationality and correctness of our solution of disentangling and manipulating the model training in the deep image features.

Moreover, we observe that this behavior is more evident for deeper features (features from Block-4 and Block-3) than for shallower ones (features from Block-2 and Block-1). An intuitive explanation is the gradient vanishing phenomenon of CNNs [17]. Due to the gradients being back-propagated, repeated multiplication and convolution with small weights render the gradient information ineffectively small in shallower blocks. Therefore, detaching the AS or PS-related gradient propagation in the shallower layers (Block-1 or Block-2) can result in a smaller impact on the model updating than the deeper layers (Block-3 or Block-4). These behaviors also guide the principle of the disentangle point selection. A latter disentangle point can achieve better performance resisting the label noise.

6.2. Training Details

In this section, we give more implementation details about our experiments. We use three kinds of synthetic label noises for CIFAR-10 and CIFAR-100: symmetric class-dependent label noise [48] (Symmetric), pairflip

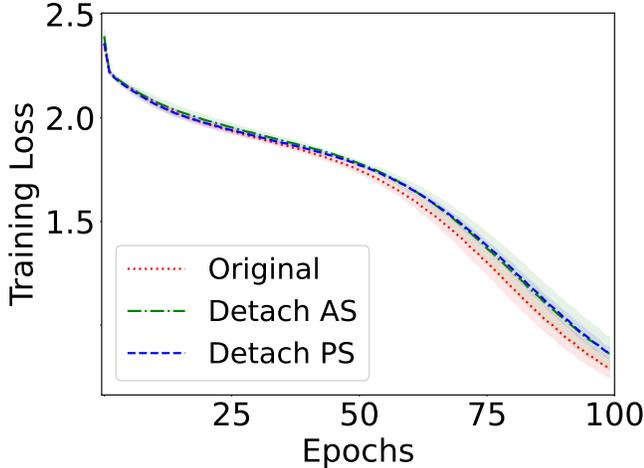
class-dependent label noise [14] (Pairflip), and instance-dependent label noise [57] (Instance). We follow the implementation of ([4, 14, 57]) to generate these label noises with different levels, which can be found in PES.

Data preprocessing For learning with confident samples (Table 1 in the paper), we apply the random crop and random horizontal flip as data augmentations. We further add MixUp [66] data augmentation for semi-supervised settings (Table 2 in the paper). For CIFAR-N dataset (Table 3 in the paper), we use random crop, random horizontal, and a CIFAR-10 augmentation policy from ([34]). The input image size of CIFAR-like datasets is set as 32×32 . For the Clothing-1M dataset (Table 4 in the paper), we first resize input images to the size of 256×256 , then randomly crop the image as 224×224 , and random horizontal flip the images last.

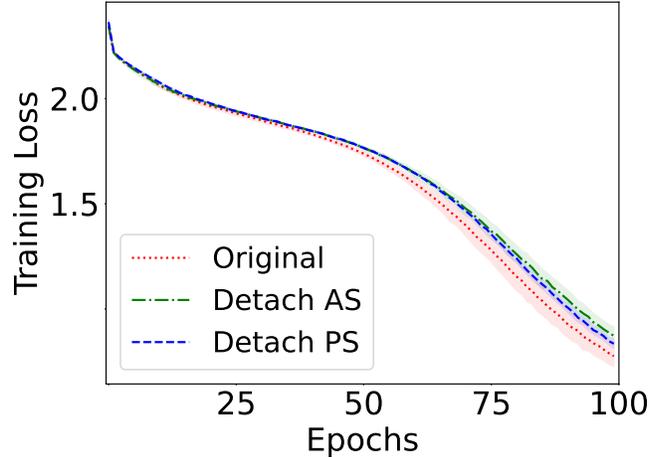
Hyper-parameters of PADDLES In learning with confident sample settings, we adopt ResNet-18 as the backbone for CIFAR-10 and ResNet-34 for CIFAR-100. We set the learning rate as 0.1, the weight decay as 10^{-4} , the batch size as 128, and the training epochs is 110. For PES training parameters, we use Adam optimizer, and set the PES learning rate is 10^{-4} , T_2, T_3 in [4] are 7 and 5 separately. Different types and levels of label noises result in different converge points of deep model on AS and PS. Therefore, we set different stopping points of T_A and T_P for different kinds and levels of label noises. For CIFAR-10, the T_A for 20%/40% Instance noise, 45% Pairflip noise, and 20%/50% Symmetric noise are [17, 20, 19, 18, 19]. The corresponding T_P are [13, 25, 16, 21, 20]. For CIFAR-100, the T_A for 20%/40% Instance noise, 45% Pairflip noise, and 20%/50% Symmetric noise are [20, 20, 19, 29, 20]. The corresponding T_P are [22, 22, 26, 11, 13]. The T_0 in Algorithm 2 is set as 0, and the training loss is the cross-entropy loss.

In semi-supervised learning, we adopt PreAct ResNet-18 as the backbone. The learning rate is 0.02 with a SGD optimizer, and we use cosine annealing learning rate scheduler to control the update of the learning rate. We set the weight decay as 5×10^{-4} , the batch size as 128, the training epochs as 500, and T_2 in [4] as 5. We train the semi-supervised models using MixMatch [5] loss with same parameters (λ_u, T, K) in [4]. Moreover, we set T_0 in Algorithm 2 as 0.

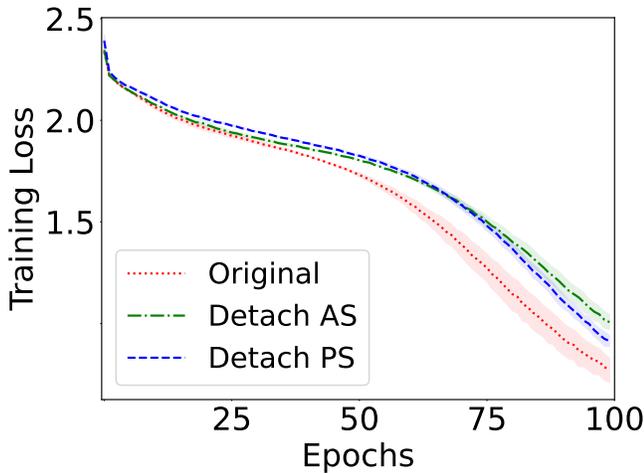
For CIFAR-N datasets, we use the ResNet-34 architecture. We set the learning rate as 0.02, the batch size as 128, the weight decay as 5×10^{-4} , the training epochs as 300, the T_2 in PES as 5. We also employ the MixMatch loss to train the semi-supervised model with MixMatch parameter λ_u as 5 and 75 for CIFAR-10N and CIFAR-100N, respectively. We set T_0 in Algorithm 2 as 1, and we do observe



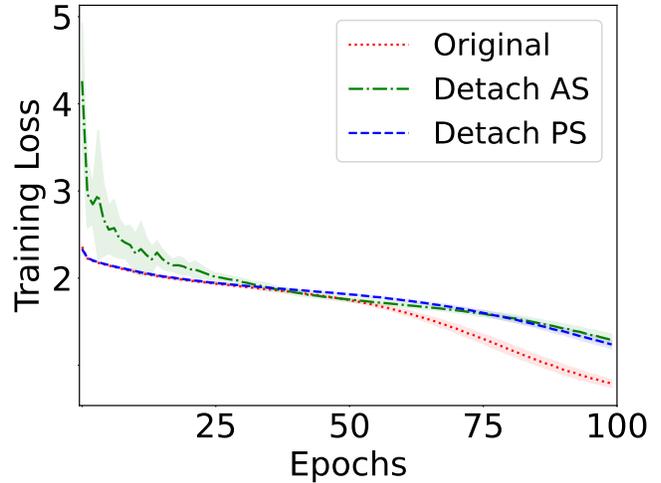
(a) ResNet Bolck-1 features under 50% Symmetric label nose



(b) ResNet Bolck-2 features under 50% Symmetric label nose



(c) ResNet Bolck-3 features under 50% Symmetric label nose



(d) ResNet Bolck-4 features under 50% Symmetric label nose

Figure 4. To evaluate the impact of Symmetric label noise on deep models with different frequency components extracted from different CNNs layers, we train a ResNet-18 model on CIFAR-10 using original image, amplitude spectrum (detach the gradients computing on phase spectrum), and phase spectrum (detach the gradients computing on phase spectrum) from different ResNet blocks. The X-axis illustrates the training epochs. Figure 4a presents the training losses of detach AS and PS components from the first ResNet block, indicated as “Detach AS” and “Detach PS” separately, and the “Original” represents train the ResNet-18 without any manipulation in the frequency domain. Figure 4b, Figure 4c, and Figure 4d show the corresponding training losses of the ResNet block 2, block 3 and block 4. The curves are based on five random experiments.

further performance improvement with a bigger T_0 like 5 in our CIFAR-N settings.

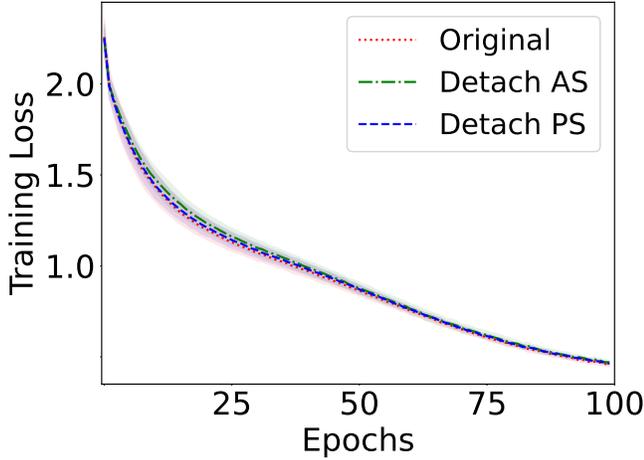
For Clothing-1M dataset, we employ the ResNet-50 as the backbone, which is pre-trained on the ImageNet. We set the batch size as 64, and the training epochs as 150. During training, we adopt the SGD optimizer with the learning rate as 4.5×10^{-3} , the weight decay as 0.001, and the momentum as 0.9. We also use a three phase OneCycle [43] scheduler to dynamic adjust the learning rate with the max learning rate as 8.55×10^{-3} . The corresponding PES learning rate is set as 5×10^{-6} and the T_2 is 7. Moreover, the

training loss is the weighted cross-entropy loss, and T_0 in Algorithm 2 is as 0. More details will be found in our scheduled released codes.

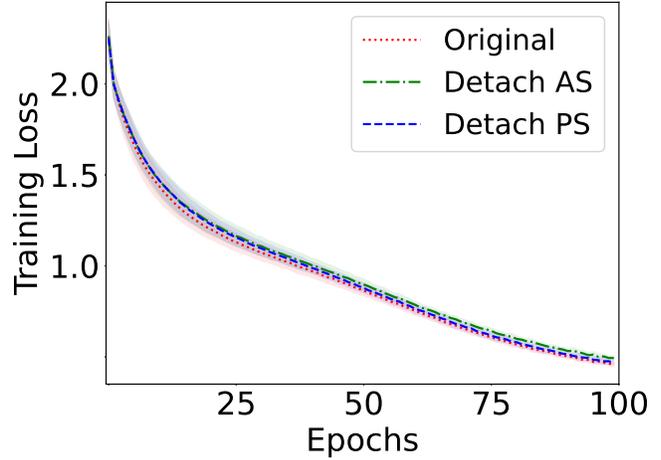
6.3. Additional Experiments

In this section, we provide more experimental results to further demonstrate the effectiveness of our methods, including training curves under different kinds of noise, confident samples quality evaluation, running time comparison, and evaluation on a text dataset.

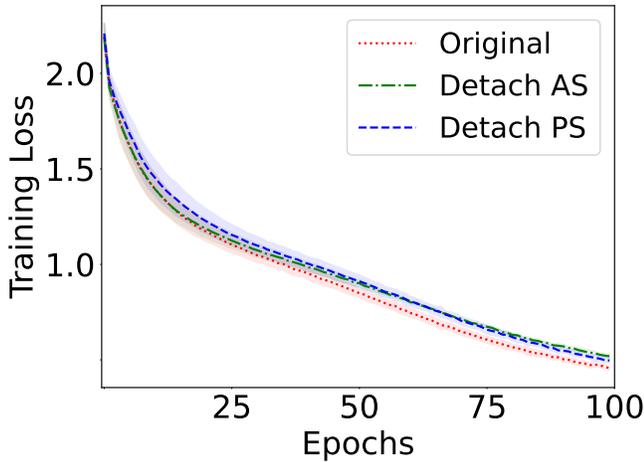
We first give more illustration about the impact of differ-



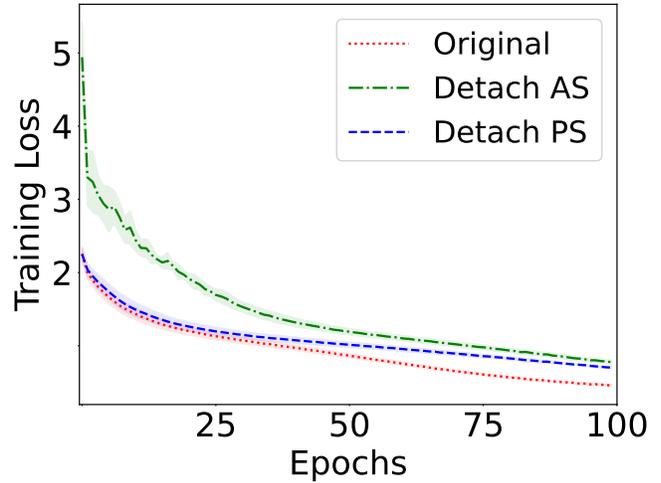
(a) ResNet Bolck-1 features under 40% Instance label nose



(b) ResNet Bolck-2 features under 40% Instance label nose



(c) ResNet Bolck-3 features under 40% Instance label nose



(d) ResNet Bolck-4 features under 40% Instance label nose

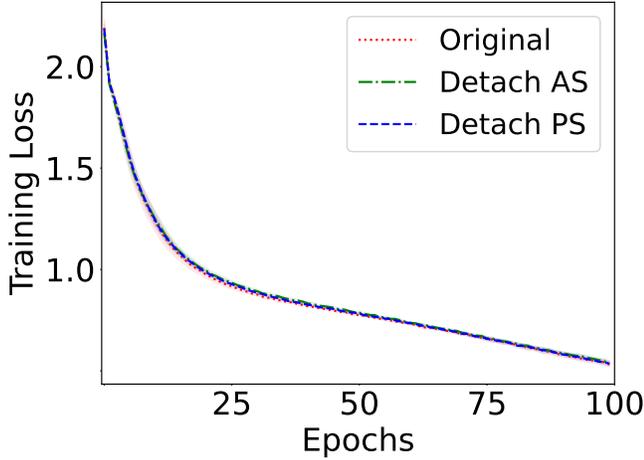
Figure 5. To evaluate the impact of Instance label noise on deep models with different frequency components extracted from different CNNs layers, we train a ResNet-18 model on CIFAR-10 using original image, amplitude spectrum (detach the gradients computing on phase spectrum), and phase spectrum (detach the gradients computing on phase spectrum) from different ResNet blocks. The X-axis illustrates the training epochs. Figure 5a presents the training losses of detach AS and PS components from the first ResNet block, indicated as “Detach AS” and “Detach PS” separately, and the “Original” represents train the ResNet-18 without any manipulation in the frequency domain. Figure 5b, Figure 5c, and Figure 5d show the corresponding training losses of the ResNet block 2, block 3 and block 4. The curves are based on five random experiments.

ent kinds of label noises on deep models in Figure 7. We generate two more kinds of label noises: the Pairflip [14] with a 45% noise rate and the Instance [57] with a 40% noise rate. As can be observed that the inflection point of AS’s loss decline is earlier than that of PS components, which means the converge speed of CNN on AS is faster than PS. Moreover, the curves of AS and PS get closer as the training epochs increase, indicating that the PS is more robust than AS with different label noises. Another evidence of the difference between AS and PS is that the number of training steps to achieve optimal performance is not

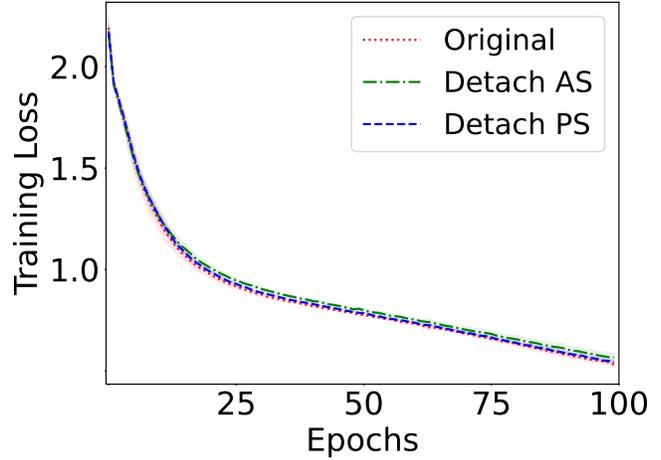
the same, and Figures 7c and 7f show that AS costs less time, achieving the best performance than PS. Both Figure 1 in our paper and Figure 7 in this material inspire us to decompose the AS and PS from the input images and design different stopping points to obtain a more robust deep network over previous ES models.

6.3.1 Confident Samples Quality

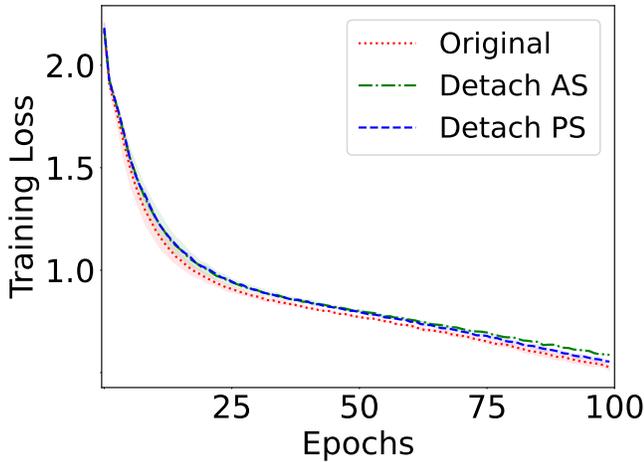
Following ([4]), we examine the extracted labels’ quality in terms of three aspects: test accuracy, label recall, and label precision using CIFAR-10, where label recall indicates the



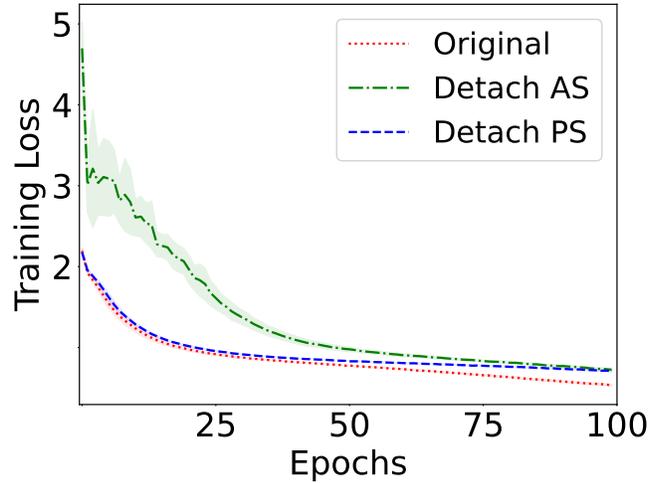
(a) ResNet Bolck-1 features under 45% Pairflip label nose



(b) ResNet Bolck-2 features under 45% Pairflip label nose



(c) ResNet Bolck-3 features under 45% Pairflip label nose



(d) ResNet Bolck-4 features under 45% Pairflip label nose

Figure 6. To evaluate the impact of Pairflip label noise on deep models with different frequency components extracted from different CNNs layers, we train a ResNet-18 model on CIFAR-10 using original image, amplitude spectrum (detach the gradients computing on phase spectrum), and phase spectrum (detach the gradients computing on phase spectrum) from different ResNet blocks. The X-axis illustrates the training epochs. Figure 6a presents the training losses of detach AS and PS components from the first ResNet block, indicated as “Detach AS” and “Detach PS” separately, and the “Original” represents train the ResNet-18 without any manipulation in the frequency domain. Figure 6b, Figure 6c, and Figure 6d show the corresponding training losses of the ResNet block 2, block 3 and block 4. The curves are based on five random experiments.

ratio of extracted confident samples with correct labels to the whole correctly labeled samples, and label precision indicates the ratio of extracted confident samples with correct labels to the whole confident samples. Specifically, we train a neural network based on ResNet-18 with various kinds and levels of label noise for total 25 epochs separately. As for our methods, the disentangle point is set between the 3rd and 4th ResNet blocks, while the stopping points of \mathcal{AS}_χ , \mathcal{PS}_χ are set to 23 and 25, respectively. The results are shown in Table 6.

From the results in Table 6, we can clearly observe that

the models generally outperform the corresponding CE and PES methods when using our methods. That is, our methods can help to obtain higher accuracy, recall, and comparable precision in the majority of cases. The collection of more confident samples is essential for learning with confident samples and semi-supervised learning. More importantly, models with high recall values can help to collect more confident samples for the following supervised or semi-supervised training. Consequently, PADDLES can contribute to improving the final classification performance in all cases by improving the performance of the initial

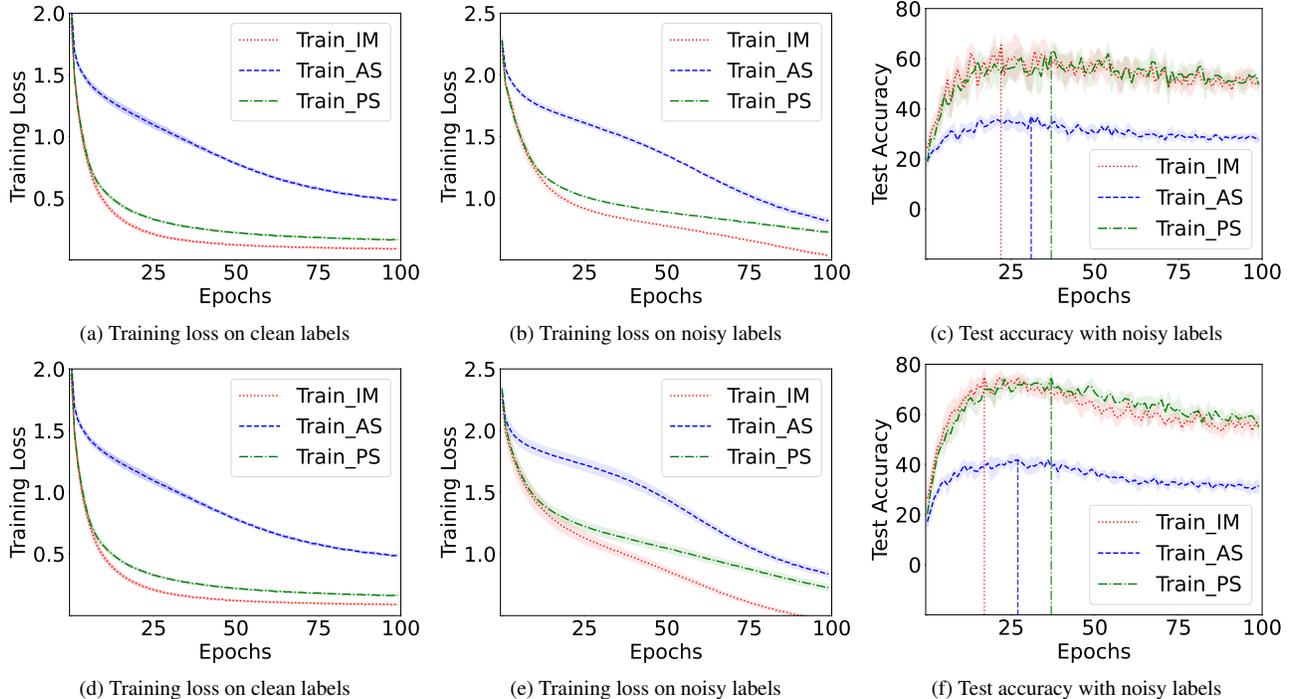


Figure 7. To evaluate the impact of label noise on deep models with different image components, we train a ResNet-18 model on CIFAR-10 using original images, amplitude spectrum, and phase spectrum under clean and noisy labels. The training losses on two kinds of labels (Figure 7a and Figure 7b 7e) and testing accuracy with the noisy labels (Figure 7c 7f) are given. The X-axis illustrates the training epochs. Figure 7b 7c are based on the 45% Pairflip label noises and Figure 7e 7f are based on the 40% Instance label noises. The curves are based on five random experiments, and the dotted vertical lines indicate the best performance steps of different image components.

model, which is also supported by the experiments in our paper.

6.3.2 Training Time Comparison

We compare the training time of proposed PADDLES and other baseline methods. For fairness, we follow [4] to conduct the experiments based on a single Nvidia V100 GPU server. Moreover, we run 200 and 300 training epochs for supervised and semi-supervised settings (noted as PADDLES(Semi)), respectively. The results are presented in Table 7. The proposed PADDLES model costs 1.55h for the supervised training, which is faster than the three methods (CDR, ELR+, and DivideMix) and achieves comparable training speed to Co-teaching. For the semi-supervised setting, due to the import of DFT, iDFT, and MixMatch training, PADDLES is slower than PES but still faster than DivideMix.

6.3.3 Text Classification

In order to further explore the generalizability of PADDLES, we also evaluate it on the text dataset NEWS. The NEWS dataset, also known as 20 Newsgroups ([20]), col-

lected by Ken Lang, is widely used as a benchmark for text classification. The original NEWS dataset contains approximately 20,000 articles among 20 classes. For fairness comparison, we follow Co-teaching+ ([64]) to re-organize the dataset with 7 classes and set 11,314 samples for training and 7,532 samples for testing. To test the extreme performance of models, we selected two difficult typical noise types with high noise rates: Symmetric 80% and Pariflip 45%.

We adopt the same network architecture of NEWS in ([64]) as the backbone to build PES-like models and our PADDLES-like models. Specifically, the backbone consists of a pretrained word embedding layer ([39]) followed by a 3-layer MLP with Softsign active function. Besides the PES, PADDLES, and their semi-supervised versions, we also extend these two ES strategies into Co-teaching frameworks, denoted as PES_Co-teaching/+ and PADDLES_Co-teaching/+ in Table 8. We empirically choose different parameters to obtain the best performance for each approach. For example, PADDLES_Co-teaching/+ adopts the PADDLES training stage to obtain good initial models for Co-teaching training, the disentangle point is set between the 2nd and 3rd layers of the MLP backbone, while the stopping points of \mathcal{AS}_x , \mathcal{PS}_x are set to 3 and 6, respectively.

Table 6. Analysis of the performance and the quality of the confident samples extracted from CIFAR-10. Mean and standard deviation over five runs are reported.

Metric	Method	Symmetric		Pairflip	Instance	
		20%	50%	45%	20%	40%
Test Accuracy	CE	82.55±2.46	70.76±1.24	60.62±5.59	84.41±0.90	74.73±2.65
	PADDLES_Base	84.73±0.65	74.34±2.06	63.68±1.59	85.63±1.16	76.70±3.60
	PES	85.87±1.59	75.87±1.33	62.40±2.34	86.58±0.45	77.07±1.18
	PADDLES	86.98±0.56	76.62±1.66	64.39±1.79	86.79±0.78	78.44±2.17
Label Recall	CE	88.51±2.26	75.18±1.00	67.84±5.06	90.37±1.01	82.15±3.17
	PADDLES_Base	91.48±0.88	79.18±2.25	70.14±3.34	91.99±0.89	84.02±4.87
	PES	92.67±1.43	81.03±1.83	71.06±2.27	93.24±0.60	85.91±0.68
	PADDLES	93.29±1.26	82.10±2.12	74.28±5.45	93.90±1.02	84.90±2.93
Label Precision	CE	98.81±0.15	94.65±0.19	72.53±5.26	98.70±0.43	90.77±1.87
	PADDLES_Base	98.83±0.08	95.01±0.27	72.97±3.01	98.52±0.26	89.83±2.73
	PES	98.96±0.09	95.46±0.14	72.99±2.27	98.52±0.19	90.63±0.92
	PADDLES	98.89±0.08	95.34±0.29	73.38±5.28	98.30±0.32	88.68±3.00

Table 7. Training time comparison for different methods on CIFAR-10 with 50% Symmetric label noise. The results of the baseline methods are taken from [4].

CE	Co-teaching	CDR	T-revision	ELR+	DivideMix	PES	PES(Semi)	Ours	Ours(Semi)
0.9h	1.5h	3.0h	3.5h	2.2h	5.5h	1.0h	3.1h	1.55h	4.8h

Table 8. Test accuracy comparison with state-of-the-art methods on the text dataset NEWS [64]. Mean and standard deviation over five runs are reported.

Method	Symmetric	Pariflip
	80%	45%
CE	19.00±0.41	31.94±0.38
PES	20.69±1.42	31.99±0.41
PADDLES	21.30±1.73	32.45±0.91
PES(Semi)	22.00±2.89	35.45±1.77
PADDLES(Semi)	22.97±4.76	35.51±1.75
Co-teaching	23.26±2.99	35.94±2.68
Co-teaching+	23.52±2.72	34.65±2.25
PES.Co-teching+	24.11±1.29	35.21±2.04
PADDLES.Co-teaching+	25.66±2.63	36.04±1.89

We train 2 models simultaneously with PADDLES, end the PADDLES training after 6 epochs, and then pass the 2 models into the Co-teaching+ network to continue the training for 20 epochs following the ways in [64]. The results are shown in Table 8.

Through the results in Table 8, we observe that the Co-teaching methods achieve superior performances over PES and PADDLES, under heavy noises, which might be caused by the difference between the text and image data. The proposed PADDLES still outperforms the baseline CE and PES models consistently. More importantly, with PADDLES pretrained base models, PADDLES.Co-teaching+ achieves the state-of-the-art among all methods. As PADDLES is

proposed from the data view, it can be combined with different LNL models and help to obtain more confidence samples. Therefore, by training with more confident samples, we can provide a more robust initial model for other subsequent models. Overall, we demonstrate the effectiveness of the proposed PADDLES for different input signals (images and texts) as well as various backbones (CNNs and MLP).