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

Huaxi Huang[1*†], Hui Kang[2†], Sheng Liu[3], Olivier Salvado[1],
Thierry Rakotoarivelo[1], Dadong Wang[1], Tongliang Liu[2‡]
[1] Data61, CSIRO, [2] The University of Sydney, [3] NYU Center for Data Science

Abstract

Convolutional Neural Networks (CNNs) are powerful in learning patterns of different vision tasks, but they are sensitive to label noise and may overfit to 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 CNNs 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. Introduction

Learning from noisy labels (LNL) [1, 34] is an active area of research within the deep learning community [45, 13, 38, 15, 67, 64, 69, 21]. Noisy labels are common in real-world applications [61, 56, 66, 49], and trustworthy AI should be robust to mislabelling.

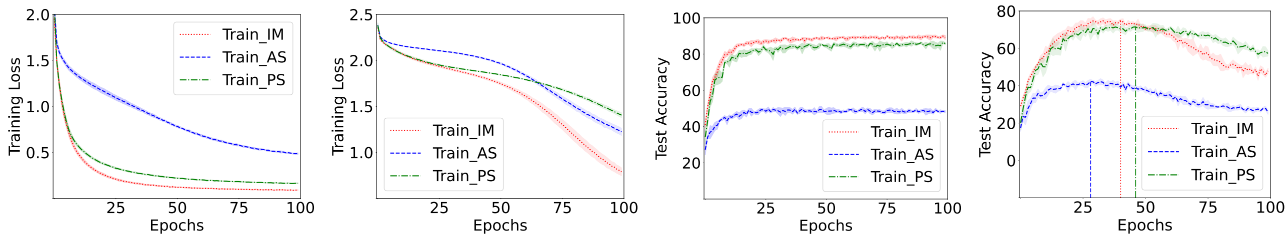
It has been argued that CNNs learn first the actual pattern before overfitting the noise [3], which inspired many works in LNL [15, 58, 27, 28, 63, 32, 31]. A training strategy is early stopping (ES), which stops the gradient-based optimization at a specific early training step. Due to its effectiveness, ES is widely applied in current LNL models and has achieved promising performance [53, 27, 39, 4, 31].

*Project lead. †Co-first authors. ‡Corresponding Author: Tongliang Liu (tongliang.liu@sydney.edu.au). Our codes are available at https://github.com/CoderHHX/PADDLES.

The frequency and spatial domains are alternative codes for depicting signal data such as images and text [42, 50]. 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 [6, 47, 14] 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, 14] indicate that the PS is the key to locating salient object areas and holds visible structured information for vision recognition [41, 12, 26], 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 [41, 14, 26, 8]. 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 [22, 57]. 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, 20]. 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 used DFT to transform raw images from CIFAR-10 [25] into AS and PS. Then, three ResNet-18 models [17] were trained using standard cross-entropy loss with raw images, AS, and PS



(a) Training loss with clean labels (b) Training loss with noisy labels (c) Testing after clean-label training (d) Testing after noisy-label training

Figure 1: Results of training ResNet-18 models on CIFAR-10 using original images, AS, and PS components ("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 noisy 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.

as inputs. Figures 1a and 1c illustrate the results of these models trained on clean labeled samples. Figure 1a shows the training losses, while Figure 1c displays the testing accuracy. The models were also trained with 50% symmetric label noise [55, 15], and the results are shown in Figures 1b and 1d, where Figure 1b shows the training loss, and Figure 1d presents the classification accuracy on the clean testing set. Under noisy labels, the convergence speeds of CNNs on AS and PS differ more than with clean labels, as shown in Figures 1a and 1b. When CNNs begin to overfit the noisy labels, they converge much faster on AS than on PS (Figure 1b). Meanwhile, the convergence speed on PS is slower than on AS and raw images, suggesting that PS can help make CNNs more robust to mislabeled data compared to 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 1d). 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 shifting to utilize more PS components.

Controlling the optimization of the model on the raw AS or PS components is difficult since current CNNs are trained based on gradient updates via backward propagation, and the raw images are fixed and do not require gradient computations during optimization. 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 noisy labels (LNL) 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 in LNL. Our method can benefit from the robustness of the PS without losing information on AS during the training of CNN. Extensive experiments on several benchmark datasets validate the effectiveness of the proposed method.

2. Related Work

2.1. Learning with noisy labels

Current methods [45, 13, 38, 43, 54, 74, 24, 15, 46, 72, 23, 71, 35, 28, 27, 19, 37, 70, 64, 69, 9, 75, 11, 44, 68, 62, 33, 60] of LNL can be grouped into two categories: model-based and model-free approaches.

Model-based methods [34, 63, 64, 70, 33] 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, [13] used a noise adaptation layer on the top of a classification model to learn the transition probabilities. T-revision [65] added fine-tuned slack variables to estimate the noise transition matrix without anchor points. Moreover, a recent work [33] 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. 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 [15, 27, 4, 63] aim to utilize the memorization effect of deep models to suppress the negative impact of the noisy labels. A representative method is Co-teaching [15], which uses two deep networks to train each other with small-loss instances in mini-batches. DivideMix [27] further extended Co-teaching with two Beta Mixture Models. Moreover, DivideMix imported MixMatch [5] training to boost the LNL 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. 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 [22, 57, 30, 8]. [57] points out that high-frequency components from an image play significant roles in improving the performance of CNNs. Moreover, [30] investigated the PS in face forgery detection and found that CNNs trained with PS can boost 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 attacks. Inspired by these breakthroughs, we are the first to investigate the frequency domain in LNL and 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 $\mathcal{D} = \{(x,y)\,|x \in \mathcal{X}, y \in \{1,\ldots,K\}\}$, where $\mathcal{X}$ is the sample space, and $\{1,\ldots,K\}$ 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 $\widehat{\mathcal{D}} = \{(x,\widehat{y})\,|x \in \mathcal{X}, \widehat{y} \in \{1,\ldots,K\}\}$ 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^*:

$$\Theta^* = \arg\min_{\Theta} \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}\left(\hat{y}_i, f_{\Theta_T} \circ f_{\Theta_{T-1}} \circ \cdots \circ f_{\Theta_0}(x_i)\right), \tag{1}$$

where $\Theta = \{\Theta_T, \Theta_{T-1}, \ldots, \Theta_0\}$ 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 $f_{\Theta_T} \circ f_{\Theta_{T-1}} \circ \cdots \circ f_{\Theta_0}(\cdot)$ since the deep neural networks can be viewed as a stack of non-linear functions. Θ_T denotes the parameter of the Tth non-linear function. $x_i, \hat{y}_i$ represent the ith sample and its label, and $\mathcal{L}$ denotes the training loss.

To obtain Θ^*, previous works [32, 63, 4] developed various optimization policies from the perspective of robust loss function design [32], gradient regulation [63], 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

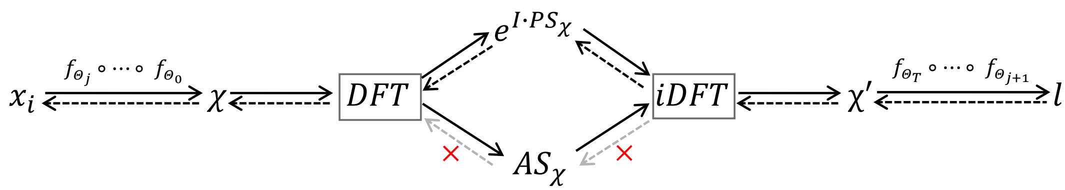


Figure 2: An illustration of the proposed PADDLES strategy stops the amplitude spectrum's involvement in model training. "⟶" denotes the forward propagation, while the "⇠" represents the backward propagation. Using Equations 2 3 and 4, we form a computational chain to disentangle the frequency domain 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.

of input data on the spatial domain [7, 42] and the vision properties of amplitude and phase spectra [6, 26], as discussed previously. Specifically, for an input sample x_i, the deep feature after jth operation in f_Θ can be represented as $\mathcal{X} = f_{\Theta_j} \circ \cdots \circ f_{\Theta_0}(x_i)$, and its frequency domain representation $\mathcal{F}_\mathcal{X}$ can be computed using DFT:

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

which can be denoted as $\mathcal{F}_\mathcal{X} = DFT(\mathcal{X})$. u represents a specific frequency, M is the number of sampled points, I is the imaginary unit, and $\mathcal{X}_p$ denotes the value at the position p of $\mathcal{X}$. 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 $\mathcal{F}_\mathcal{X}(u)$ is a complex-valued variable, its real part can be denoted as $Real_{\mathcal{F}_\mathcal{X}}$, and the imaginary part is $Imag_{\mathcal{F}_\mathcal{X}}$. We then disentangle the phase and amplitude components using the following rules:

$$\begin{aligned} \mathcal{PS}_\mathcal{X}(u) &= \arctan(\frac{Imag_{\mathcal{F}_\mathcal{X}}(u)}{Real_{\mathcal{F}_\mathcal{X}}(u)}), \\ \mathcal{AS}_\mathcal{X}(u) &= |\mathcal{F}_\mathcal{X}(u)|, \end{aligned} \tag{3}$$

where $\mathcal{PS}_\mathcal{X}$ represents the phase spectrum, $\mathcal{AS}_\mathcal{X}$ 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:

$$\mathcal{X}'_p = \frac{1}{M} \sum_{u=0}^{M-1} (e^{I \cdot PS_\mathcal{X}(u)} \odot AS_\mathcal{X}(u)) e^{\frac{I \cdot 2\pi}{M} pu}, \tag{4}$$

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

Through Equation 2, 3, and 4, we construct a computation flow disentangling the phase spectrum $\mathcal{PS}_\mathcal{X}$ and the amplitude spectrum $\mathcal{AS}_\mathcal{X}$ from the original feature $\mathcal{X}$ 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 propagation. The forward propagation (right arrows in Figure 2) will generate the intermediate values $(\mathcal{X}, \mathcal{PS}_\mathcal{X}, \mathcal{AS}_\mathcal{X}, \mathcal{X}')$ 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. Through Equation 3, we obtain the $PS_\mathcal{X}$ and $AS_\mathcal{X}$ from $\mathcal{F}_\mathcal{X}$. We denote $r = e^{I \cdot PS_\mathcal{X}} \odot AS_\mathcal{X}$ in Equation 4. During the back-propagation, we need to compute the partial derivatives of the loss function $\mathcal{L}$ with respect to $\mathcal{F}_\mathcal{X}$, *i.e.*, $\frac{\partial \mathcal{L}}{\partial \mathcal{F}_\mathcal{X}}$. Using the chain rule, $\frac{\partial \mathcal{L}}{\partial \mathcal{F}_\mathcal{X}} = \frac{\partial \mathcal{L}}{r} \cdot \left(\frac{\partial e^{I \cdot PS_\mathcal{X}}}{\partial \mathcal{F}_\mathcal{X}} \odot AS\mathcal{X} + e^{I \cdot PS_\mathcal{X}} \odot \frac{\partial AS_\mathcal{X}}{\partial \mathcal{F}_\mathcal{X}} \right)$. For PADDLES, detaching the "AS components" means stopping the computation of $\frac{\partial AS_\mathcal{X}}{\partial \mathcal{F}_\mathcal{X}}$, resulting in: $\frac{\partial \mathcal{L}}{\partial \mathcal{F}_\mathcal{X}} = \frac{\partial \mathcal{L}}{r} \cdot \left(\frac{\partial e^{I \cdot PS_\mathcal{X}}}{\partial \mathcal{F}_\mathcal{X}} \odot AS\mathcal{X} \right)$. Similarly, detaching the "PS components" means stopping the computation of $\frac{\partial e^{I \cdot PS_\mathcal{X}}}{\partial \mathcal{F}_\mathcal{X}}$, resulting in: $\frac{\partial \mathcal{L}}{\partial \mathcal{F}_\mathcal{X}} = \frac{\partial \mathcal{L}}{r} \cdot \left(e^{I \cdot PS\mathcal{X}} \odot \frac{\partial AS_\mathcal{X}}{\partial \mathcal{F}_\mathcal{X}} \right)$. 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 part.

3.3. Practical Implementation

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

To reduce the difficulty of implementation and further improve the robustness of PADDLES, we incorporate 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 $\{\Theta_0^*, \ldots, \Theta_j^*\}$ are well-optimized. We then apply PES to update the remaining parts $\{\Theta_{j+1}, \ldots, \Theta_T\}$ with previous

Algorithm 1: Model Update with AS/PS control

Input : A noisy set $\hat{\mathcal{D}}$, Disentangle point j, Deep Model $f_{\Theta=\{\Theta_T,\Theta_{T-1},\ldots,\Theta_0\}}$, Target Spectrum ($\mathcal{AS}_\chi$ or $\mathcal{PS}_\chi$).

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

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

Algorithm 2: PADDLES

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

for $i=1$ **to** T_A **do**
  Update model parameter Θ using Equation 1;
end
for $i=1$ **to** T_P **do**
  Update model parameter using Algorithm 1 with $\mathcal{AS}_\chi$ detached;
end
for $i=1$ **to** T_0 **do**
  Update model parameter using Algorithm 1 with $\mathcal{PS}_\chi$ detached;
end
Hook $\mathcal{AS}_\chi$ and $\mathcal{PS}_\chi$ to the gradient computation graph during backpropagation;
for $l=j+1$ **to** T **do**
  Freeze $\{\Theta_0,\ldots,\Theta_{l-1}\}$ and re-initialize other parameters;
  for $i=1$ **to** T_l **do**
    Update network parameter $\{\Theta_l,\ldots,\Theta_T\}$ using Equation 5;
  end
end

Output: The optimized model f_{Θ^*}.

parameters fixed. T_l steps will be performed during training using the following objective:

$$\min_{\{\Theta_{l,..,T}\}} \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}\left(\hat{y}_i, f_{\Theta_T}\circ..\circ f_{\Theta_l}\circ f_{\Theta^*_{l-1}}\circ..\circ f_{\Theta^*_0}(x_i)\right),$$
$$l=j+1,j+2,\cdots,T-1,T. \tag{5}$$

After the optimization with Equation 5, the final model $f_{\Theta^*=\{\Theta^*_0,\cdots,\Theta^*_T\}}$ is obtained.

Learning Settings We adopt PADDLES as a confident sample selector to boost noisy label learning with supervised and semi-supervised learning settings. The confident sample set $\mathcal{D}_{lb}$ is defined as

$$\begin{aligned}\mathcal{D}_{lb} &= \{(x_i,\hat{y}_i)|\hat{y}_i=\bar{y}_i, i=1,\cdots,N\},\\ \bar{y}_i &= \arg\max_{\tau\in\{1,\cdots,K\}}\frac{1}{2}[f^\tau_{\Theta^*}(\mathrm{A}(x_i))+f^\tau_{\Theta^*}(\mathrm{A}'(x_i))],\end{aligned} \tag{6}$$

where A and A$'$ are data augmentation operators randomly sampled from the same augmentation set, $f^\tau_{\Theta^*}(x_i)$ 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 training.

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

$$\begin{aligned}\mathcal{D}_{ub} &= \{x_i|\hat{y}_i\neq\bar{y}_i, i=1,\cdots,N\},\\ \bar{y}_i &= \arg\max_{\tau\in\{1,\cdots,K\}}\frac{1}{2}[f^\tau_{\Theta^*}(\mathrm{A}(x_i))+f^\tau_{\Theta^*}(\mathrm{A}'(x_i))].\end{aligned} \tag{7}$$

We adopt the MixMatch [5] loss in the semi-supervised learning as previous works [27, 4].

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 [25], and two real-world noisy sets: CIFAR-N [60] and Clothing-1M [66]. 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 [15, 32, 63, 65]. 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 crawled 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 [73] data augmentation for semi-supervised settings. For CIFAR-N dataset, we use

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**

a CIFAR-10 augmentation policy from [40]. 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 of 256×256, then randomly crop the image to 224×224, and horizontally flip the images with a random probability.

Following [27, 4], we use 10% of the dataset with noisy labels as the validation set for CIFAR, and we also use the official clean validate set from Clothing-1M in our work. Hyper-parameters are tuned based on the validation set.

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 [73] strategy, respectively. 2) Classic LNL methods: Co-teaching [15], Forward-T [43], JointOptim [53], T-revision [65], M-correction [2], DMI [67] and JoCoR [59]. 3) State-of-the art LNL methods: DivideMix [27], CDR [63], ELR [32], PES [4], CORES [9] and SOP [33].

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, re-

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 [60]. 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**

spectively. 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 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 $\mathcal{AS}_\chi$, $\mathcal{PS}_\chi$ 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 [48] 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 $\mathcal{AS}_\chi$, $\mathcal{PS}_\chi$ 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 outperforms 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 model on a more challenging real-world noise-label dataset, as shown in Table 3. CIFAR-

Table 5: Ablation studies of 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

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

Components Study: 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, PADDLES_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, PADDLES gains further improvement and achieves state-of-the-art performance since the proposed training strategy is designed from the view of the data frequency domain, which is orthogonal to the PES.

Disentangle Position: Another study 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. 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 the disentangle position between the 3rd and 4th ResNet blocks. More details are delineated in *Supplemental Materials Section 2*.

Hyperparameter Sensitivity: We investigate the hyperparameter sensitivity of the early stopping points for both the amplitude spectrum T_A and the phase spectrum T_P. All experiments are conducted on CIFAR-N datasets with a ResNet-34 backbone. We vary T_A from 18 to 30 with $T_P = 5$ and set T_P from 5 to 17 with $T_A = 30$. It is observed that with fixed T_P, the performance of PADDLES will generally increase when T_A is growing for both Fine noises on CIFAR-100N and Worst noises on CIFAR-10N. When 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 performance of our model on CIFAR-10N dataset with Aggregate noise stays comparatively stable in comparison with other noises. The model achieves the best performance with $T_A = 30$ and $T_P = 5$. More illustrations about the hyperparameter sensitivity analysis can be found in *Appendix*.

Table 6: Comparison of Training time 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+
0.9h	1.5h	3.0h	3.5h	2.2h
DivideMix	PES	PES(Semi)	Ours	Ours(Semi)
5.5h	1.0h	3.1h	1.55h	4.8h

Complexity and Training Time: PADDLES employs the operations DFT and iDFT, the additional complexity of PADDLES is $\mathcal{O}(B \times C \times HW \times log(B \times C \times HW))$, where B, C, HW denote the batch size, channel size, and feature-map size. To quantitatively measure the additional computational cost, we compare the training time of the proposed PADDLES and other methods. For fairness, we follow [4] to conduct the experiments based on a single Nvidia V100 GPU server. We train the model 200 and 300 epochs under the supervised and semi-supervised settings (noted as Ours(Semi)), respectively. The results are presented in Table 6. The training takes 1.55h for the supervised training, which is faster than the three methods (CDR, ELR+, and DivideMix) and achieves comparable computational performance to Co-teaching. For the semi-supervised setting, PADDLES is slower than PES but faster than DivideMix.

PADDLES with CNN Architecture: We investigated the effectiveness of our proposed PADDLES model using different vision architectures. Firstly, we explored the deeper ResNet models, ResNet-101 and ResNet-152 from [18]. The models were trained for 200 epochs using the SGD optimizer, with an initial learning rate of 0.1, weight decay of 10^{-4}, and batch size of 128. The disentangling point of PADDLES was set between the 3rd and 4th ResNet blocks. The updated epochs of PES and T_A, T_P were the same as the shallow ResNet experiments presented in Table 1 of the main paper. As demonstrated in Table 7, deeper ResNet backbones resulted in higher performances for CE methods. Importantly, PADDLES consistently outperformed PES and CE on different ResNet models and CIFAR sets.

Besides ResNet models, we investigated the effectiveness of our proposed method using a modern vision architecture EfficientNet [51, 52]. We utilized the PyTorch Efficient B0 model. The models were trained for 200 epochs using the Adam optimizer with an initial learning rate of

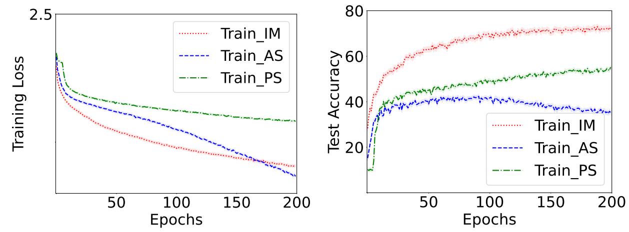
Table 7: Ablation studies of the proposed PADDLES using different backbones. All methods are trained with 50% Symmetric label noise. Mean and standard deviation computed over five independent runs are reported.

Dataset	Method	ResNet-101	ResNet-152	EfficientNet
CIFAR-10	CE	77.22±1.54	78.16±0.43	75.59±0.94
	PES	86.12±0.19	86.41±0.54	77.49±0.68
	PADDLES	87.13±0.41	87.85±0.28	78.90±0.62
CIFAR-100	CE	37.01±2.11	37.05±1.65	42.49±0.57
	PES	57.12±1.94	57.28±1.72	45.53±1.06
	PADDLES	57.72±2.24	58.07±1.28	46.14±0.84

10^{-3}, weight decay of 10^{-4}, and batch size of 128. EfficientNet B0 has nine blocks in its feature encoder, which differs from the ResNet model used in our previous experiment. For CIFAR-10, we set the disentangling point of PADDLES between the 5th and 6th EfficientNet blocks, with $T_A = 129$ and $T_P = 7$. For CIFAR-100, we set the disentangling point between the 6th and 7th EfficientNet blocks, with $T_A = 125$ and $T_P = 7$. We used the PES update strategy described in the original paper [4]. For CIFAR-10, we re-trained the 6th, 7th, 8th, 9th EfficientNet blocks and the last fully-connected layer with 20, 15, 7, 5, and 5 epochs. For CIFAR-100, we refined the 7th, 8th, and 9th EfficientNet blocks and the last fully-connected layer with 15, 7, 5, and 5 epochs. The results in Table 7 demonstrate that PADDLES outperformed CE and PES under different EfficientNet models and datasets.

PADDLES with Vision Transformer: Vision Transformer architecture has demonstrated effectiveness on various computer vision tasks [10, 36, 16]. PADDLES builds upon the established roles of AS and PS in CNNs [22, 57, 30]. However, the roles of AS and PS in Transformer-based models have received comparatively less attention in the current literature. Consequently, while PADDLES has been demonstrated to be effective in conjunction with CNNs, its efficacy with Transformer models may require further theoretical investigation. In this section, instead of applying PADDLES to Transformer directly, we first conduct an experimental study to examine the behavior of the Transformer trained with AS and PS components under label noise.

To investigate the impact of label noise on Transformer models trained on different image components, we selected Swin Transformer [36] (Swin_S) as our backbone. We applied DFT to transform raw images from CIFAR-10 [25] into AS and PS. Subsequently, we trained three Swin_S models using standard cross-entropy loss with raw images, AS, and PS as inputs. We trained the models with 50% symmetric label noise, using the Adam optimizer with an initial learning rate of 3×10^{-4}, a weight decay of 10^{-4}, and a batch size of 128. The results are illustrated in Figures 3a and 3b, where Figure 3a shows the training loss, and Figure 3b displays the classification accuracy on the clean testing set. Under noisy labels, Swin Transformer's convergence speeds on AS and PS differ, as depicted in Figures 3a. It converges faster on AS when overfitting noisy labels. Thus, PS might enhance Vision Transformer's resistance to label noise compared to AS, a trend also seen in CNN backbones. Given the limited research on Vision Transformers' robustness in the frequency domain, our findings offer an initial insight, hoping to spur further exploration.



(a) Training loss with noisy labels (b) Testing after noisy-label training

Figure 3: Results of training Swin Transformers on CIFAR-10 using original images, AS, and PS on noisily labeled sets. The curves are averaged across five random runs.

After the above study, we tested PADDLES against CE on the Swin_S backbone. The training lasted 200 epochs with the Adam optimizer, a learning rate of 3×10^{-4}, and weight decay of 10^{-4}. The feature encoder of Swin_S has eight blocks, and we set the disentangling point between the 7th and 8th blocks of Swin_S's eight blocks with $T_A = 127$ and $T_P = 7$. PADDLES surpassed CE by 2.13% (75.07% vs. 72.94%), proving its efficacy on the Vision Transformer.

More experiments for PADDLES: We further validated PADDLES through additional experiments, including an experiment on a large-scale Webvision 1.0 [29] dataset. Details are provided in the *Supplemental Materials*.

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 demonstrate the effectiveness of the PADDLES.

Acknowledgment

Tongliang Liu is partially supported by the following Australian Research Council projects: FT220100318, DP220102121, LP220100527, LP220200949, and IC190100031. Huaxi Huang is supported by the CSIRO R+ CERC Fellow Project: Differential privacy preservation machine learning on image data.

References

[1] Dana Angluin and Philip Laird. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988.

[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.

[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.

[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.

[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.

[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.

[7] Kenneth R Castleman. *Digital image processing*. Prentice Hall Press, 1996.

[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.

[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.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[11] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, and Chiyuan Zhang. Deep learning with label differential privacy. *NeurIPS*, 34:27131–27145, 2021.

[12] Dennis C Ghiglia and Mark D Pritt. Two-dimensional phase unwrapping: theory, algorithms, and software. *A Wiley Interscience Publication*, 1998.

[13] Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. *ICLR*, 2017.

[14] 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.

[15] 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.

[16] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on vision transformer. *IEEE transactions on pattern analysis and machine intelligence*, 45(1):87–110, 2022.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, June 2016.

[18] 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.

[19] Wei Hu, Zhiyuan Li, and Dingli Yu. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. In *ICLR*, 2020.

[20] Zhuo Huang, Xiaobo Xia, Li Shen, Bo Han, Mingming Gong, Chen Gong, and Tongliang Liu. Harnessing out-of-distribution examples via augmenting content and style. In *ICLR*, 2023.

[21] Zhuo Huang, Miaoxi Zhu, Xiaobo Xia, Li Shen, Jun Yu, Chen Gong, Bo Han, Bo Du, and Tongliang Liu. Robust generalization against photon-limited corruptions via worst-case sharpness minimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16175–16185, 2023.

[22] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *NeurIPS*, 32, 2019.

[23] 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.

[24] Jan Kremer, Fei Sha, and Christian Igel. Robust active label correction. In *AISTATS*, pages 308–316. PMLR, 2018.

[25] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

[26] 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.

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

[28] 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.

[29] Wen et al. Li. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017.

[30] 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.

[31] Sheng Liu, Kangning Liu, Weicheng Zhu, Yiqiu Shen, and Carlos Fernandez-Granda. Adaptive early-learning correction for segmentation from noisy annotations. *CVPR*, 2022.

[32] Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *NeurIPS*, 2020.

[33] Sheng Liu, Zhihui Zhu, Qing Qu, and Chong You. Robust training under label noise by over-parameterization. In *ICML*, 2022.

[34] Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on pattern analysis and machine intelligence*, 38(3):447–461, 2015.

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

[36] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[37] Yueming Lyu and Ivor W Tsang. Curriculum loss: Robust learning and generalization against label corruption. In *ICLR*, 2020.

[38] Eran Malach and Shai Shalev-Shwartz. Decoupling" when to update" from" how to update". *NeurIPS*, 30, 2017.

[39] 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.

[40] Kento Nishi, Yi Ding, Alex Rich, and Tobias Hollerer. Augmentation strategies for learning with noisy labels. In *CVPR*, pages 8022–8031, 2021.

[41] Alan V Oppenheim and Jae S Lim. The importance of phase in signals. *Proceedings of the IEEE*, 69(5):529–541, 1981.

[42] Alan V Oppenheim, Alan S Willsky, Syed Hamid Nawab, Gloria Mata Hernández, et al. *Signals & systems*. Pearson Educación, 1997.

[43] 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.

[44] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *NeurIPS*, 2021.

[45] 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.

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

[47] Eero P Simoncelli and Odelia Schwartz. Modeling surround suppression in v1 neurons with a statistically derived normalization model. *NeurIPS*, pages 153–159, 1999.

[48] 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.

[49] 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.

[50] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[51] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[52] Mingxing Tan and Quoc Le. Efficientnetv2: Smaller models and faster training. In *International conference on machine learning*, pages 10096–10106. PMLR, 2021.

[53] Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *CVPR*, pages 5552–5560, 2018.

[54] Kiran K Thekumparampil, Ashish Khetan, Zinan Lin, and Sewoong Oh. Robustness of conditional gans to noisy labels. *NeurIPS*, 31, 2018.

[55] Brendan Van Rooyen, Aditya Menon, and Robert C Williamson. Learning with symmetric label noise: The importance of being unhinged. *NeurIPS*, 28, 2015.

[56] Sudheendra Vijayanarasimhan and Kristen Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. *IJCV*, 108(1):97–114, 2014.

[57] 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.

[58] 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.

[59] Hongxin Wei, Lei Feng, Xiangyu Chen, and Bo An. Combating noisy labels by agreement: A joint training method with co-regularization. In *CVPR*, 2020.

[60] 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.

[61] Peter Welinder and Pietro Perona. Online crowdsourcing: rating annotators and obtaining cost-effective labels. In *CVPR Workshops*, pages 25–32. IEEE, 2010.

[62] 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.

[63] 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.

[64] 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.

[65] 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.

[66] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *CVPR*, 2015.

[67] Yilun Xu, Peng Cao, Yuqing Kong, and Yizhou Wang. L_dmi: A novel information-theoretic loss function for training deep nets robust to label noise. *NeurIPS*, 2019.

[68] Erkun Yang, Dongren Yao, Tongliang Liu, and Cheng Deng. Mutual quantization for cross-modal search with noisy labels. In *CVPR*, pages 7551–7560, 2022.

[69] 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.

[70] 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.

[71] 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.

[72] Xiyu Yu, Tongliang Liu, Mingming Gong, and Dacheng Tao. Learning with biased complementary labels. In *ECCV*, pages 68–83, 2018.

[73] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

[74] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *NeurIPS*, 31, 2018.

[75] Zhaowei Zhu, Tongliang Liu, and Yang Liu. A second-order approach to learning with instance-dependent label noise. In *CVPR*, pages 10113–10123, 2021.