Learning From Noisy Labels With Deep Neural Networks: A Survey

Hwanjun Song, Minseok Kim, *Student Member, IEEE*, Dongmin Park,
Yooju Shin, and Jae-Gil Lee, *Member, IEEE*

***Abstract*— Deep learning has achieved remarkable success in numerous domains with help from large amounts of big data. However, the quality of data labels is a concern because of the lack of high-quality labels in many real-world scenarios. As noisy labels severely degrade the generalization performance of deep neural networks, learning from noisy labels (robust training) is becoming an important task in modern deep learning applications. In this survey, we first describe the problem of learning with label noise from a supervised learning perspective. Next, we provide a comprehensive review of 62 state-of-the-art robust training methods, all of which are categorized into five groups according to their methodological difference, followed by a systematic comparison of six properties used to evaluate their superiority. Subsequently, we perform an in-depth analysis of noise rate estimation and summarize the typically used evaluation methodology, including public noisy datasets and evaluation metrics. Finally, we present several promising research directions that can serve as a guideline for future studies.**

***Index Terms*— Classification, deep learning, label noise, noisy label, robust deep learning, robust optimization, survey.**

I. Introduction

WITH the recent emergence of large-scale datasets, deep neural networks (DNNs) have exhibited impressive performance in numerous machine learning tasks, such as computer vision [1], [2], information retrieval [3]–[5], and language processing [6]–[8]. Their success is dependent on the availability of massive but carefully labeled data, which are expensive and time-consuming to obtain. Some nonexpert sources, such as Amazon's Mechanical Turk and the surrounding text of collected data, have been widely used to mitigate the high labeling cost; however, the use of these sources often results in unreliable labels [9]–[12]. In addition, data labels can be extremely complex even for experienced domain experts [13], [14]; they can also be adversarially manipulated by a label-flipping attack [15]. Such unreliable labels are called *noisy labels* because they may be *corrupted* from ground-truth labels. The ratio of corrupted labels in real-world datasets is reported to range from 8.0% to 38.5% [16]–[19].

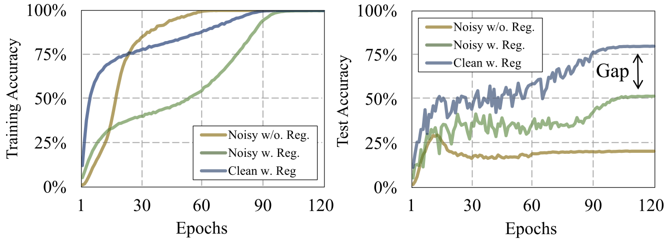

Fig. 1. Convergence curves of training and test accuracy when training WideResNet-16-8 using a standard training method on the CIFAR-100 dataset with the symmetric noise of 40%: "Noisy w/o. Reg." and "Noisy w. Reg." are the models trained on noisy data without and with regularization, respectively, and "Clean w. Reg." is the model trained on clean data with regularization.

In the presence of noisy labels, training DNNs is known to be susceptible to noisy labels because of the significant number of model parameters that render DNNs overfit to even corrupted labels with the capability of learning any complex function [20], [21]. Zhang *et al.* [22] demonstrated that DNNs can easily fit an entire training dataset with any ratio of corrupted labels, which eventually resulted in poor generalizability on a test dataset. Unfortunately, popular regularization techniques, such as data augmentation [23], weight decay [24], dropout [25], and batch normalization [26] have been applied extensively, but they do *not* completely overcome the overfitting issue by themselves. As shown in Fig. 1, the gap in test accuracy between models trained on clean and noisy data remains significant even though all of the aforementioned regularization techniques are activated. In addition, the accuracy drop with label noise is considered to be more harmful than with other noises, such as input noise [27]. Hence, achieving a good generalization capability in the presence of noisy labels is a key challenge.

Several studies have been conducted to investigate supervised learning under noisy labels. Beyond conventional machine learning techniques [13], [28], deep learning techniques have recently gained significant attention in the machine learning community. In this survey, we present the advances in recent deep learning techniques for overcoming noisy labels. We surveyed 174 recent studies by recursively tracking relevant bibliographies in papers published at premier research conferences, such as CVPR, ICCV, NeurIPS, ICML,

Manuscript received 15 July 2020; revised 6 April 2021 and 10 November 2021; accepted 15 February 2022. Date of publication 7 March 2022; date of current version 30 October 2023. This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2020-0-00862, DB4DL: High-Usability and Performance In-Memory Distributed DBMS for Deep Learning). *(Corresponding author: Jae-Gil Lee.)*

Hwanjun Song is with the NAVER AI Laboratory, Seongnam 13561, South Korea (e-mail: ghkswns91@gmail.com).

Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee are with the Graduate School of Knowledge Service Engineering, Korea Advanced Institute of Science and Technology, Daejeon 34141, South Korea (e-mail: minseokkim@kaist.ac.kr; dongminpark@kaist.ac.kr; yooju24@kaist.ac.kr; jaegil@kaist.ac.kr).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3152527.

Digital Object Identifier 10.1109/TNNLS.2022.3152527

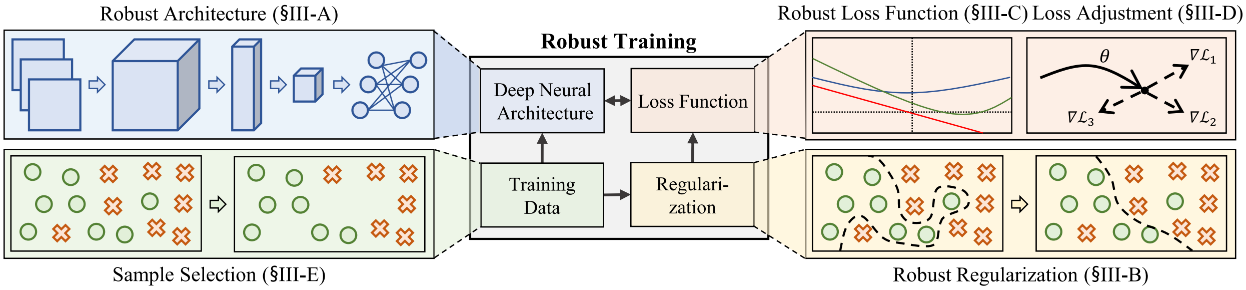

Fig. 2. Categorization of recent deep learning methods for overcoming noisy labels.

and ICLR. Although we attempted to comprehensively include all recent studies at the time of submission, some of them may not be included because of the quadratic increase in deep learning papers. The studies included were grouped into *five* categories, as shown in Fig. 2 (see Section III for details).

A. Related Surveys

Frénay and Verleysen [13] discussed the potential negative consequence of learning from noisy labels and provided a comprehensive survey on noise-robust classification methods, focusing on conventional supervised approaches, such as naïve Bayes and support vector machines. Furthermore, their survey included the definitions and sources of label noise and the taxonomy of label noise. Zhang *et al.* [28] discussed another aspect of label noise in crowdsourced data annotated by nonexperts and provided a thorough review of expectation–maximization (EM) algorithms that were proposed to improve the quality of crowdsourced labels. Meanwhile, Nigam *et al.* [29] provided a brief introduction to deep learning algorithms that were proposed to manage noisy labels; however, the scope of these algorithms was limited to only two categories, i.e., the loss function and sample selection in Fig. 2. Recently, Han *et al.* [30] summarized the essential components of robust learning with noisy labels, but their categorization is totally different from ours in philosophy; we mainly focus on systematic methodological difference, whereas they rather focused on more general views, such as input data, objective functions, and optimization policies. Furthermore, this survey is the first to present a comprehensive methodological comparison of existing robust training approaches.

B. Survey Scope

Robust training with DNNs becomes critical to guarantee the reliability of machine learning algorithms. In addition to label noise, two types of flawed training data have been actively studied by different communities [31], [32]. *Adversarial learning* is designed for small, worst case perturbations of the inputs, so-called adversarial examples, which are maliciously constructed to deceive an already trained model into making errors [33]–[36]. Meanwhile, *data imputation* primarily deals with missing inputs in training data, where missing values are estimated from the observed ones [32], [37]. Adversarial learning and data imputation are closely related to robust learning, but handling *feature* noise is beyond the scope of this survey—i.e., learning from noisy *labels*.

TABLE I

SUMMARY OF THE NOTATION

Notation	Description
$\mathcal{X}$	the data feature space
$\mathcal{Y}$, $\tilde{\mathcal{Y}}$	the true and noisy label space
$\mathcal{D}$, $\tilde{\mathcal{D}}$	the clean and noisy training data
$P_{\mathcal{D}}$, $P_{\tilde{\mathcal{D}}}$	the joint distributions of clean and noisy data
$\mathcal{B}_t$	a set of mini-batch examples at time t
Θ_t	the parameter of a deep neural network at time t
$f(\cdot;\Theta_t)$	a deep neural network parameterized by Θ_t
ℓ	a specific loss function
$\mathcal{R}$	an empirical risk
$\mathbb{E}_{\mathcal{D}}$	an expectation over $\mathcal{D}$
x, x_i	a data example of $\mathcal{X}$
y, y_i	a true label of $\mathcal{Y}$
$\tilde{y}$, $\tilde{y}_i$	a noisy label of $\tilde{\mathcal{Y}}$
η	a specific learning rate
τ	a true noise rate
b	the number of mini-batch examples in $\mathcal{B}_t$
c	the number of classes
T, $\hat{\text{T}}$	the true and estimated noise transition matrix

II. PRELIMINARIES

In this section, the problem statement for supervised learning with noisy labels is provided along with the taxonomy of label noise. Managing noisy labels is a long-standing issue; therefore, we review the basic conventional approaches and theoretical foundations underlying robust deep learning. Table I summarizes the notation frequently used in this study.

A. Supervised Learning With Noisy Labels

Classification is a representative supervised learning task for learning a function that maps an input feature to a label [38]. In this article, we consider a c-class classification problem using a DNN with a softmax output layer. Let $\mathcal{X} \subset \mathbb{R}^d$ be the feature space and $\mathcal{Y} = \{0,1\}^c$ be the ground-truth label space in a *one-hot* manner. In a typical classification problem, we are provided with a training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ obtained from an unknown joint distribution $P_{\mathcal{D}}$ over $\mathcal{X} \times \mathcal{Y}$, where each (x_i, y_i) is *independent and identically distributed*. The goal of the task is to learn the mapping function $f(\cdot;\Theta) : \mathcal{X} \to [0,1]^c$ of the DNN parameterized by Θ such that the parameter Θ minimizes the empirical risk $\mathcal{R}_{\mathcal{D}}(f)$

$$\mathcal{R}_{\mathcal{D}}(f) = \mathbb{E}_{\mathcal{D}}\big[\ell\big(f(x;\Theta), y\big)\big] = \frac{1}{|\mathcal{D}|}\sum_{(x,y)\in\mathcal{D}} \ell\big(f(x;\Theta), y\big) \tag{1}$$

where ℓ is a certain loss function.

As data labels are corrupted in various real-world scenarios, we aim to train the DNN from noisy labels. Specifically, we are provided with a noisy training dataset $\tilde{\mathcal{D}} = \{(x_i, \tilde{y}_i)\}_{i=1}^{N}$ obtained from a noisy joint distribution $P_{\tilde{\mathcal{D}}}$ over $\mathcal{X} \times \tilde{\mathcal{Y}}$, where $\tilde{y}$ is a *noisy* label which may not be true. Hence, following the standard training procedure, a mini-batch $\mathcal{B}_t = \{(x_i, \tilde{y}_i)\}_{i=1}^{b}$ comprising b examples is obtained randomly from the noisy training dataset $\tilde{\mathcal{D}}$ at time t. Subsequently, the DNN parameter Θ_t at time t is updated along the descent direction of the empirical risk on mini-batch $\mathcal{B}_t$

$$\Theta_{t+1} = \Theta_t - \eta \nabla \left(\frac{1}{|\mathcal{B}_t|} \sum_{(x,\tilde{y}) \in \mathcal{B}_t} \ell\big(f(x; \Theta_t), \tilde{y}\big) \right) \tag{2}$$

where η is a learning rate specified.

Here, the risk minimization process is no longer *noise-tolerant* because of the loss computed by the noisy labels. DNNs can easily memorize corrupted labels and correspondingly degenerate their generalizations on unseen data [13], [28], [29]. Hence, mitigating the adverse effects of noisy labels is essential to enable noise-tolerant training for deep learning.

B. Taxonomy of Label Noise

This section presents the types of label noise that have been adopted to design robust training algorithms. Even if data labels are corrupted from ground-truth labels without *any* prior assumption, in essence, the corruption probability is affected by the dependency between *data features* or *class labels*. A detailed analysis of the taxonomy of label noise was provided by Frénay and Verleysen [13]. Most existing algorithms dealt with instance-independent noise, but instance-dependent noise has not yet been extensively investigated owing to its complex modeling.

1) Instance-Independent Label Noise: A typical approach for modeling label noise assumes that the corruption process is conditionally *independent* of data features when the true label is given [22], [39]. That is, the true label is corrupted by a *noise transition matrix* $\mathrm{T} \in [0, 1]^{c \times c}$, where $\mathrm{T}_{ij} := p(\tilde{y} = j | y = i)$ is the probability of the true label i being flipped into a corrupted label j. In this approach, the noise is called a *symmetric* (or *uniform*) noise with a noise rate $\tau \in [0, 1]$ if $\forall_{i=j}\mathrm{T}_{ij} = 1 - \tau \wedge \forall_{i \neq j}\mathrm{T}_{ij} = (\tau/(c-1))$, where a true label is flipped into other labels with equal probability. In contrast to symmetric noise, the noise is called an *asymmetric* (or *label-dependent*) noise if $\forall_{i=j}\mathrm{T}_{ij} = 1 - \tau \wedge \exists_{i \neq j, i \neq k, j \neq k}\mathrm{T}_{ij} > \mathrm{T}_{ik}$, where a true label is more likely to be mislabeled into a particular label. For example, a "dog" is more likely to be confused with a "cat" than with a "fish." In a stricter case when $\forall_{i=j}\mathrm{T}_{ij} = 1 - \tau \wedge \exists_{i \neq j}\mathrm{T}_{ij} = \tau$, the noise is called a *pair noise*, where a true label is flipped into only a certain label.

2) Instance-Dependent Label Noise: For more realistic noise modeling, the corruption probability is assumed to be *dependent* on both the data features and class labels [16], [40]. Accordingly, the corruption probability is defined as $\rho_{ij}(x) = p(\tilde{y} = j | y = i, x)$. Unlike the aforementioned noises, the data feature of an example x also affects the chance of x being mislabeled.

C. Nondeep Learning Approaches

For decades, numerous methods have been proposed to manage noisy labels using conventional machine learning techniques. These methods can be categorized into *four* groups [13], [29], [41], as follows.

1) *Data Cleaning:* Training data are cleaned by excluding examples whose labels are likely to be corrupted. Bagging and boosting are used to filter out false-labeled examples to remove examples with higher weights because false-labeled examples tend to exhibit much higher weights than true-labeled examples [42], [43]. In addition, various methods, such as k-nearest neighbor, outlier detection, and anomaly detection, have been widely exploited to exclude false-labeled examples from noisy training data [44]–[46]. Nevertheless, this family of methods suffers from an over-cleaning issue that overly removes even the true-labeled examples.
2) *Surrogate Loss:* Motivated by the noise-tolerance of the 0–1 loss function [39], many researchers have attempted to resolve its inherent limitations, such as computational hardness and nonconvexity that render gradient methods unusable. Hence, several convex surrogate loss functions, which approximate the 0–1 loss function, have been proposed to train a specified classifier under the binary classification setting [47]–[51]. However, these loss functions cannot support the multiclass classification task.
3) *Probabilistic Method:* Under the assumption that the distribution of features is helpful in solving the problem of learning from noisy labels [52], the confidence of each label is estimated by clustering and then used for a weighted training scheme [53]. This confidence is also used to convert hard labels into soft labels to reflect the uncertainty of labels [54]. In addition to these clustering approaches, several Bayesian methods have been proposed for graphical models such that they can benefit from using any type of prior information in the learning process [55]. However, this family of methods may exacerbate the overfitting issue owing to the increased number of model parameters.
4) *Model-Based Method:* As conventional models, such as the SVM and decision tree, are not robust to noisy labels, significant effort has been expended to improve their robustness. To develop a robust SVM model, misclassified examples during learning are penalized in the objective [56], [57]. In addition, several decision tree models are extended using new split criteria to solve the overfitting issue when the training data are not fully reliable [58], [59]. However, it is infeasible to apply their design principles to deep learning.

Meanwhile, deep learning is more susceptible to label noises than traditional machine learning owing to its high expressive power, as proven by many researchers [21], [60], [61]. There has been significant effort to understand why noisy labels negatively affect the performance of DNNs [22], [61]–[63]. This theoretical understanding has led to the algorithmic design which achieves higher robustness than nondeep

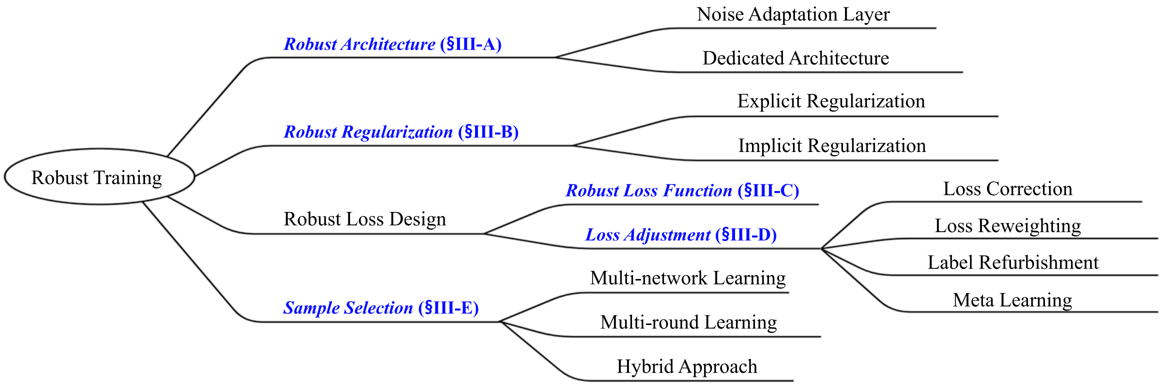

Fig. 3. High-level research overview of robust deep learning for noisy labels. The research directions that are actively contributed by the machine learning community are categorized into five groups in blue italic.

learning methods. A detailed analysis of theoretical understanding for robust deep learning was provided by Han *et al.* [30].

D. Regression With Noisy Labels

In addition to classification, regression is another main topic of supervised machine learning, which aims to model the relationship between a number of features and a continuous target variable. Unlike the classification task with a *discrete* label space, the regression task considers the continuous variable as its target label [64], and thus, it learns the mapping function $f(\,\cdot\,;\Theta) : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{Y} \in \mathbb{R}$ is a *continuous* label space. Given the input feature x and its ground-truth label y, two types of label noise are considered in the regression task. An *additive noise* [65] is formulated by $\tilde{y} := y + \epsilon$, where ϵ is drawn from a random distribution independent of the input feature; an *instance-dependent noise* [66] is formulated by $\tilde{y} := \rho(x)$ where $\rho : \mathcal{X} \rightarrow \mathcal{Y}$ is a noise function dependent on the input feature.

Although regression predicts continuous values, regression and classification share the same concept of learning the mapping function from the input feature x to the output label y. Thus, many robust approaches for classification are easily extended to the regression problem with simple modification [67]. Thus, in this survey, we focus on the classification setting for which the most robust methods are defined.

III. Deep Learning Approaches

According to our comprehensive survey, the robustness of deep learning can be enhanced in numerous approaches [16], [25], [68]–[74]. Fig. 3 shows an overview of recent research directions conducted by the machine learning community. All of them (i.e., Sections III-A–III-E) focused on making a supervised learning process more robust to label noise.

1) Robust Architecture (Section III-A): Adding a noise adaptation layer at the top of an underlying DNN to learn label transition process or developing a dedicated architecture to reliably support more diverse types of label noise.
2) Robust Regularization (Section III-B): Enforcing a DNN to overfitting less to false-labeled examples explicitly or implicitly.
3) Robust Loss Function (Section III-C): Improving the loss function.
4) Loss Adjustment (Section III-D): Adjusting the loss value according to the confidence of a given loss (or label) by loss correction, loss reweighting, or label refurbishment.
5) Sample Selection (Section III-E): Identifying true-labeled examples from noisy training data via multinetwork or multiround learning.

Overall, we categorize all recent deep learning methods into *five* groups corresponding to popular research directions, as shown in Fig. 3. In Section III-D, meta-learning is also discussed because it finds the optimal hyperparameters for loss reweighting. In Section III-E, we discuss the recent efforts for combining sample selection with other orthogonal directions or semisupervised learning toward the state-of-the-art performance.

Fig. 2 shows the categorization of robust training methods using these five groups.

A. Robust Architecture

In numerous studies, architectural changes have been made to model the noise transition matrix of a noisy dataset [16], [75]–[82]. These changes include adding a noise adaptation layer at the top of the softmax layer and designing a new dedicated architecture. The resulting architectures yield improved generalization through the modification of the DNN output based on the estimated label transition probability.

1) Noise Adaptation Layer: From the view of training data, the noise process is modeled by discovering the underlying label transition pattern (i.e., the noise transition matrix T). Given an example x, the noisy class posterior probability for an example x is expressed by

$$p(\tilde{y} = j|x) = \sum_{i=1}^{c} p(\tilde{y} = j, y = i|x) = \sum_{i=1}^{c} \mathrm{T}_{ij}\, p(y = i|x)$$
$$\text{where } \mathrm{T}_{ij} = p(\tilde{y} = j|y = i, x). \tag{3}$$

In light of this, the noise adaptation layer is intended to mimic the label transition behavior in learning a DNN. Let $p(y|x;\Theta)$ be the output of the base DNN with a softmax output layer. Then, following (3), the probability of an

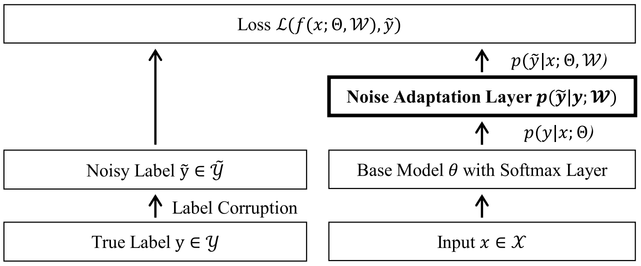
Loss $\mathcal{L}(f(x;\Theta,\mathcal{W}),\tilde{y})$
$p(\tilde{y}|x;\Theta,\mathcal{W})$
Noise Adaptation Layer $p(\tilde{y}|y;\mathcal{W})$
$p(y|x;\Theta)$
Noisy Label $\tilde{y}\in\tilde{\mathcal{Y}}$
Base Model θ with Softmax Layer
Label Corruption
True Label $y\in\mathcal{Y}$
Input $x\in\mathcal{X}$

Fig. 4. Noise modeling process using the noise adaptation layer.

example x being predicted as its noisy label $\tilde{y}$ is parameterized by

$$\begin{aligned} p(\tilde{y}=j|x;\Theta,\mathcal{W}) &= \sum_{i=1}^{c} p(\tilde{y}=j, y=i|x;\Theta,\mathcal{W}) \\ &= \sum_{i=1}^{c} \underbrace{p(\tilde{y}=j|y=i;\mathcal{W})}_{\text{Noise Adaptation Layer}} \underbrace{p(y=i|x;\Theta)}_{\text{Base Model}}. \end{aligned} \tag{4}$$

Here, the noisy label $\tilde{y}$ is assumed to be *conditionally independent* of the input x in general. Accordingly, as shown in Fig. 4, the noisy adaptation layer is added at the top of the base DNN to model the noise transition matrix parameterized by $\mathcal{W}$. This layer should be removed when test data are to be predicted.

Technical Detail: Webly learning [75] first trains the base DNN only for easy examples retrieved by search engines; subsequently, the confusion matrix for all training examples is used as the initial weight $\mathcal{W}$ of the noise adaptation layer. It fine-tunes the entire model in an end-to-end manner for hard training examples. In contrast, the *noise model* [77] initializes $\mathcal{W}$ to an identity matrix and adds a regularizer to force $\mathcal{W}$ to diffuse during DNN training. The *dropout noise model* [25] applies dropout regularization to the adaptation layer, whose output is normalized by the softmax function to implicitly diffuse $\mathcal{W}$. The *s-model* [79] is similar to the *dropout noise model* but dropout is not applied. The *c-model* [79] is an extension of the s-model that models the instance-dependent noise, which is more realistic than the symmetric and asymmetric noises. Meanwhile, *NLNN* [76] adopts the EM algorithm to iterate the E-step to estimate the noise transition matrix and the M-step to backpropagate the DNN.

Remark: A common drawback of this family is their inability to identify false-labeled examples, treating all the examples equally. Thus, the estimation error for the transition matrix is generally large when only noisy training data is used or when the noise rate is high [83]. Meanwhile, for the EM-based method, becoming stuck in local optima is inevitable, and high computational costs are incurred [79].

2) Dedicated Architecture: Beyond the label-dependent label noise, several studies have been conducted to support more complex noise, leading to the design of dedicated architectures [16], [80], [81]. They typically aimed at increasing the reliability of estimating the label transition probability to handle more complex and realistic label noise.

Technical Detail: Probabilistic noise modeling [16] manages two independent networks, each of which is specialized to predict the noise type and label transition probability. Because an EM-based approach with random initialization is impractical for training the entire network, both networks are trained with massive noisy labeled data after the pretraining step with a small amount of clean data. Meanwhile, *masking* [80] is a human-assisted approach to convey the human cognition of invalid label transitions. Considering that noisy labels are mainly from the interaction between humans and tasks, the invalid transition investigated by humans was leveraged to constrain the noise modeling process. Owing to the difficulty in specifying the explicit constraint, a variant of generative adversarial networks (GANs) [84] was employed in this study. Recently, the *contrastive-additive noise network* [81] was proposed to adjust incorrectly estimated label transition probabilities by introducing a new concept of quality embedding, which models the trustworthiness of noisy labels. *RoG* [85] builds a simple yet robust generative classifier on top of any discriminative DNN pretrained on noisy data.

Remark: Compared with the noise adaptation layer, this family of methods significantly improves the robustness to more diverse types of label noise, but it cannot be easily extended to other architectures in general.

B. Robust Regularization

Regularization methods have been widely studied to improve the generalizability of a learned model in the machine learning community [23]–[26]. By avoiding overfitting in model training, the robustness to label noise improves with widely-used regularization techniques, such as *data augmentation* [23], *weight decay* [24], *dropout* [25], and *batch normalization* [26]. These canonical regularization methods operate well on moderately noisy data, but they alone do *not sufficiently* improve the test accuracy; poor generalization could be obtained when the noise is heavy [86]. Thus, more advanced regularization techniques have been recently proposed, which further improved robustness to label noise when used along with the canonical methods. The main advantage of this family is its *flexibility* in collaborating with other directions because it only requires simple modifications.

1) Explicit Regularization: The regularization can be an explicit form that modifies the expected training loss, e.g., weight decay and dropout.

Technical Detail: Bilevel learning [87] uses a clean validation dataset to regularize the overfitting of a model by introducing a bilevel optimization approach, which differs from the conventional one in that its regularization constraint is also an optimization problem. Overfitting is controlled by adjusting the weights on each mini-batch and selecting their values such that they minimize the error on the validation dataset. Meanwhile, *annotator confusion* [86] assumes the existence of multiple annotators and introduces a regularized EM-based approach to model the label transition probability; its regularizer enables the estimated transition probability to converge to the true confusion matrix of the annotators.

In contrast, *pretraining* [88] empirically proves that fine-tuning on a pretrained model provides a significant improvement in robustness compared with models trained from scratch; the universal representations of pretraining prevent the model parameters from being updated in the wrong direction by noisy labels. *PHuber* [89] proposes a composite loss-based gradient clipping, which is a variation of standard gradient clipping for label noise robustness. *Robust early learning* [90] classifies critical parameters and noncritical parameters for fitting clean and noise labels, respectively. Then, it penalizes only the noncritical ones with a different update rule. *ODLN* [91] leverages open-set auxiliary data and prevents the overfitting of noisy labels by assigning random labels to the open-set examples, which are uniformly sampled from the label set.

Remark: The explicit regularization often introduces sensitive model-dependent hyperparameters or requires deeper architectures to compensate for the reduced capacity, yet it can lead to significant performance gain if they are optimally tuned.

2) Implicit Regularization: The regularization can also be an implicit form that gives the effect of stochasticity, e.g., data augmentation and mini-batch stochastic gradient descent.

Technical Detail: Adversarial training [92] enhances the noise tolerance by encouraging the DNN to correctly classify both original inputs and hostilely perturbed ones. *Label smoothing* [93], [94] estimates the marginalized effect of label noise during training, thereby reducing overfitting by preventing the DNN from assigning a full probability to noisy training examples. Instead of the one-hot label, the noisy label is mixed with a uniform mixture over all possible labels

$$\bar{y} = \big\langle \bar{y}(1), \bar{y}(2), \ldots, \bar{y}(c) \big\rangle$$
$$\text{where } \bar{y}(i) = (1-\alpha)\cdot[\tilde{y}=i] + \alpha/c \text{ and } \alpha \in [0,1]. \tag{5}$$

Here, $[\cdot]$ is the Iverson bracket and α is the smoothing degree. In contrast, *mixup* [95] regularizes the DNN to favor simple linear behaviors in between training examples. First, the mini-batch is constructed using virtual training examples, each of which is formed by the linear interpolation of two noisy training examples $(x_i, \tilde{y}_i)$ and $(x_j, \tilde{y}_j)$ obtained at random from noisy training data $\tilde{\mathcal{D}}$

$$x_{\text{mix}} = \lambda x_i + (1-\lambda)x_j \quad \text{and} \quad y_{\text{mix}} = \lambda \tilde{y}_i + (1-\lambda)\tilde{y}_j \tag{6}$$

where $\lambda \in [0,1]$ is the balance parameter between two examples. Thus, *mixup* extends the training distribution by updating the DNN for the constructed mini-batch.

Remark: The implicit regularization improves the generalization capability of the DNN without reducing the representational capacity. It also does not introduce sensitive model-dependent hyperparameters because it is applied to the training data. However, the extended feature or label space slows down the convergence of training.

C. Robust Loss Function

It was proven that a learned DNN with a *suitably modified* loss function ℓ' for noisy data $\tilde{\mathcal{D}}$ can approach the Bayes optimal classifier f^*, which achieves the optimal Bayes risk $\mathcal{R}^* = \mathcal{R}_{\mathcal{D}}(f^*)$ for clean data $\mathcal{D}$. Let $\hat{f} = \text{argmin}_{f\in\mathcal{F}}\hat{\mathcal{R}}_{\ell',\tilde{\mathcal{D}}}(f)$ be the learned classifier with the modified loss ℓ' for the noisy data, where $\hat{\mathcal{R}}_{\ell',\tilde{\mathcal{D}}}(f) = \mathbb{E}_{\tilde{\mathcal{D}}}[\ell(f(x;\Theta),\tilde{y})]$. If ℓ is L-Lipschitz and classification calibrated [50], with probability at least $1-\delta$, there exists a nondecreasing function ζ_ℓ with $\zeta_\ell(0) = 0$ [39] such that

$$\mathcal{R}_{\mathcal{D}}(\hat{f}) - \mathcal{R}^* \le \zeta_\ell\Big(\overbrace{\min\nolimits_{f\in\mathcal{F}}\mathcal{R}_{\ell,\mathcal{D}}(f) - \min\nolimits_f \mathcal{R}_{\ell,\mathcal{D}}(f)}^{\text{Approximation and Estimation Errors}} + 4L_p\text{RC}(\mathcal{F}) + 2\sqrt{\log(1/\delta)\big/2|\mathcal{D}|}\Big). \tag{7}$$

L_p is the Lipschitz constant of ℓ' and RC is the Rademacher complexity of the hypothesis class $\mathcal{F}$. Then, by the universal approximation theorem [96], the Bayes optimal classifier f^* is guaranteed to be in the hypothesis class $\mathcal{F}$ with DNNs.

Based on this theoretical foundation, researchers have attempted to design robust loss functions such that they achieve a small risk for unseen clean data even when noisy labels exist in the training data [68], [97]–[101].

Technical Detail: Initially, Manwani and Sastry [48] theoretically proved a sufficient condition for the loss function such that risk minimization with that function becomes noise-tolerant for binary classification. Subsequently, the sufficient condition was extended for multiclass classification using deep learning [68]. Specifically, a loss function is defined to be *noise-tolerant* for a c-class classification under *symmetric* noise if the function satisfies the noise rate $\tau < ((c-1)/c)$ and

$$\sum_{j=1}^{c} \ell\big(f(x;\Theta), y=j\big) = C \quad \forall x \in \mathcal{X}\ \forall f \tag{8}$$

where C is a constant. This condition guarantees that the classifier trained on noisy data has the same misclassification probability as that trained on noise-free data under the specified assumption. An extension for *multilabel* classification was provided by Kumar *et al.* [102]. Moreover, if $\mathcal{R}_{\mathcal{D}}(f^*) = 0$, then the function is also noise-tolerant under an *asymmetric* noise, where f^* is a global risk minimizer of $\mathcal{R}_{\mathcal{D}}$.

For the classification task, the categorical cross entropy (CCE) loss is the most widely used loss function owing to its fast convergence and high generalization capability. However, in the presence of noisy labels, the *robust MAE* [68] showed that the mean absolute error (MAE) loss achieves better generalization than the CCE loss because only the MAE loss satisfies the aforementioned condition. A limitation of the MAE loss is that its generalization performance degrades significantly when complicated data are involved. Hence, the *generalized cross entropy* (GCE) [97] was proposed to achieve the advantages of both MAE and CCE losses; the GCE loss is a more general class of noise-robust loss that encompasses both of them. Amid *et al.* [103] extended the GCE loss by introducing two temperatures based on the Tsallis divergence. *Bitempered loss* [104] introduces a proper unbiased generalization of the CE loss based on the Bregman divergence. In addition, inspired by the symmetricity of the Kullback–Leibler divergence, the symmetric cross entropy (SCE) [98] was proposed by combining a noise tolerance term, namely reverse cross entropy loss, with the standard CCE loss.

Meanwhile, the *curriculum loss* (CL) [99] is a surrogate loss of the 0–1 loss function; it provides a tight upper bound and can easily be extended to multiclass classification. The *active passive loss* (APL) [105] is a combination of two types of robust loss functions, an active loss that maximizes the probability of belonging to the given class and a passive loss that minimizes the probability of belonging to other classes.

Remark: The robustness of these methods is theoretically supported well. However, they perform well only in simple cases, when learning is easy or the number of classes is small [106]. Moreover, the modification of the loss function increases the training time for convergence [97].

D. Loss Adjustment

Loss adjustment is effective for reducing the negative impact of noisy labels by adjusting the loss of all training examples before updating the DNN [19], [62], [69], [107]–[111]. The methods associated with it can be categorized into three groups depending on their adjustment philosophy: 1) *loss correction* that estimates the noise transition matrix to correct the forward or backward loss; 2) *loss reweighting* that imposes different importance to each example for a weighted training scheme; 3) *label refurbishment* that adjusts the loss using the refurbished label obtained from a convex combination of noisy and predicted labels; and 4) *meta learning* that automatically infers the optimal rule for loss adjustment. Unlike the robust loss function newly designed for robustness, this family of methods aims to make the traditional optimization process robust to label noise. Hence, in the middle of training, the update rule is adjusted such that the negative impact of label noise is minimized.

In general, loss adjustment allows for a *full exploration* of the training data by adjusting the loss of every example. However, the error incurred by *false* correction is accumulated, especially when the number of classes or the number of mislabeled examples is large [112].

1) Loss Correction: Similar to the noise adaptation layer presented in Section III-A, this approach modifies the loss of each example by multiplying the estimated label transition probability by the output of a specified DNN. The main difference is that the learning of the transition probability is decoupled from that of the model.

Technical Detail: Backward correction [62] initially approximates the noise transition matrix using the softmax output of the DNN trained without loss correction. Subsequently, it retrains the DNN while correcting the original loss based on the estimated matrix. The corrected loss of an example $(x, \tilde{y})$ is computed by a linear combination of its loss values for observable labels, whose coefficient is the inverse transition matrix T^{-1} to the observable label $y \in \{1, \ldots, c\}$, given its target label $\tilde{y}$. Therefore, the backward correction $\overleftarrow{\ell}$ is performed by multiplying the inverse transition matrix to the prediction for all the observable labels

$$\overleftarrow{\ell}\big(f(x;\Theta), \tilde{y}\big) = \hat{\mathrm{T}}^{-1}\Big\langle \ell\big(f(x;\Theta), 1\big), \ldots, \ell\big(f(x;\Theta), c\big)\Big\rangle^{\top} \tag{9}$$

where $\hat{\mathrm{T}}$ is the estimated noise transition matrix.

Conversely, *forward correction* [62] uses a linear combination of a DNN's softmax outputs before applying the loss function. Hence, the forward correction $\overrightarrow{\ell}$ is performed by multiplying the estimated transition probability with the softmax outputs during the forward propagation step

$$\begin{aligned}\overrightarrow{\ell}\big(f(x;\Theta), \tilde{y}\big) &= \ell\Big(\Big\langle \hat{p}(\tilde{y}|1), \ldots, \hat{p}(\tilde{y}|c)\Big\rangle f(x;\Theta)^{\top}, \tilde{y}\Big) \\ &= \ell\big(\hat{\mathrm{T}}^{\top} f(x;\Theta)^{\top}, \tilde{y}\big).\end{aligned} \tag{10}$$

Furthermore, *gold loss correction* [107] assumes the availability of clean validation data or anchor points for loss correction. Thus, a more accurate transition matrix is obtained by using them as additional information, which further improves the robustness of the loss correction. Recently, *T-revision* [113] provides a solution that can infer the transition matrix without anchor points, and *dual T* [114] factorizes the matrix into the product of two easy-to-estimate matrices to avoid directly estimating the noisy class posterior. Beyond the instance-independent noise assumption, Zhang and Sugiyama [115] introduced the instance-confidence embedding to model instance-dependent noise in estimating the transition matrix. On the other hand, Yang *et al.* [116] proposed to use the Bayes optimal transition matrix estimated from the distilled examples for the instance-dependent noise transition matrix.

Remark: The robustness of these approaches is highly dependent on how precisely the transition matrix is estimated. To acquire such a transition matrix, they require prior knowledge in general, such as anchor points or clean validation data.

2) Loss Reweighting: Inspired by the concept of importance reweighting [117], loss reweighting aims to assign smaller weights to the examples with false labels and greater weights to those with true labels. Accordingly, the reweighted loss on the mini-batch $\mathcal{B}_t$ is used to update the DNN

$$\Theta_{t+1} = \Theta_t - \eta\nabla\left(\frac{1}{|\mathcal{B}_t|}\sum_{(x,\tilde{y})\in\mathcal{B}_t}\overbrace{w(x,\tilde{y})\ell\big(f(x;\Theta_t), \tilde{y}\big)}^{\text{Reweighted Loss}}\right) \tag{11}$$

where $w(x, \tilde{y})$ is the weight of an example x with its noisy label $\tilde{y}$. Hence, the examples with smaller weights do not significantly affect the DNN learning.

Technical Detail: In *importance reweighting* [108], the ratio of two joint data distributions $w(x, \tilde{y}) = P_{\mathcal{D}}(x, \tilde{y})/P_{\tilde{\mathcal{D}}}(x, \tilde{y})$ determines the contribution of the loss of each noisy example. An approximate solution to estimate the ratio was developed because the two distributions are difficult to determine from noisy data. Meanwhile, *active bias* [109] emphasizes uncertain examples with inconsistent label predictions by assigning their prediction variances as the weights for training. *Dual-Graph* [118] employs graph neural networks and reweights the examples according to the structural relations among labels, eliminating the abnormal noise examples.

Remark: These approaches need to manually prespecify the weighting function and additional hyperparameters, which is fairly hard to be applied in practice due to the significant variation of appropriate weighting schemes that rely on the noise type and training data.

3) Label Refurbishment: Refurbishing a noisy label $\tilde{y}$ effectively prevents overfitting to false labels. Let $\hat{y}$ be the current prediction of the DNN $f(x;\Theta)$. Therefore, the refurbished label y^{refurb} can be obtained by a convex combination of the noisy label $\tilde{y}$ and the DNN prediction $\hat{y}$

$$y^{\text{refurb}} = \alpha\tilde{y} + (1-\alpha)\hat{y} \tag{12}$$

where $\alpha \in [0,1]$ is the label confidence of $\tilde{y}$. To mitigate the damage of incorrect labeling, this approach backpropagates the loss for the refurbished label instead of the noisy one, thereby yielding substantial robustness to noisy labels.

Technical Detail: Bootstrapping [69] is the first method that proposes the concept of label refurbishment to update the target label of training examples. It develops a more coherent network that improves its ability to evaluate the consistency of noisy labels, with the label confidence α obtained via cross validation. *Dynamic bootstrapping* [110] dynamically adjusts the confidence α of individual training examples. The confidence α is obtained by fitting a two-component and 1-D beta mixture model to the loss distribution of all training examples. *Self-adaptive training* [119] applies the exponential moving average to alleviate the instability issue of using instantaneous prediction of the current DNN

$$y_{t+1}^{\text{refurb}} = \alpha y_t^{\text{refurb}} + (1-\alpha)\hat{y}, \quad \text{where } y_0^{\text{refurb}} = \tilde{y}. \tag{13}$$

D2L [111] trains a DNN using a dimensionality-driven learning strategy to avoid overfitting to false labels. A simple measure called *local intrinsic dimensionality* [120] is adopted to evaluate the confidence α in considering that the overfitting is exacerbated by dimensional expansion. Hence, refurbished labels are generated to prevent the dimensionality of the representation subspace from expanding at a later stage of training. Recently, *SELFIE* [19] introduces a novel concept of *refurbishable examples* that can be corrected with high precision. The key idea is to consider the example with consistent label predictions as refurbishable because such consistent predictions correspond to its true label with a high probability owing to the learner's perceptual consistency. Accordingly, the labels of only refurbishable examples are corrected to minimize the number of falsely corrected cases. Similarly, *AdaCorr* [121] selectively refurbishes the label of noisy examples, but a theoretical error bound is provided. Alternatively, *SEAL* [122] averages the softmax output of a DNN on each example over the whole training process, then retrains the DNN using the averaged soft labels.

Remark: Differently from loss correction and reweighting, all the noisy labels are explicitly replaced with other expected clean labels (or their combination). If there are not many confusing classes in data, these methods work well by refurbishing the noisy labels with high precision. In the opposite case, the DNN could overfit wrongly refurbished labels.

4) Meta Learning: In recent years, meta-learning becomes an important topic in the machine learning community and is applied to improve noise robustness [123]–[125]. The key concept is *learning to learn* that performs learning at a level higher than conventional learning, thus achieving data-agnostic and noise type-agnostic rules for better practical use. It is similar to loss reweighting and label refurbishment, but the adjustment is automated in a meta-learning manner.

Technical Detail: For the loss reweighting in (11), the goal is to learn the weight function $w(x,\tilde{y})$. Specifically, *L2LWS* [126] and *CWS* [127] are unified neural architectures composed of a target DNN and a meta-DNN. The meta-DNN is trained on a small clean validation dataset; it then provides guidance to evaluate the weight score for the target DNN. Here, part of the two DNNs is shared and jointly trained to benefit from each other. *Automatic reweighting* [106] is a meta-learning algorithm that learns the weights of training examples based on their gradient directions. It includes a small clean validation dataset into the training dataset and reweights the backward loss of the mini-batch examples such that the updated gradient minimizes the loss of this validation dataset. *Meta-weight-net* [124] parameterizes the weighting function as a multilayer perceptron network with only one hidden layer. A meta-objective is defined to update its parameters such that they minimize the empirical risk of a small clean dataset. At each iteration, the parameter of the target network is guided by the weight function updated via the meta-objective. Likewise, *data coefficients* (i.e., exemplar weights and true labels) [128] are estimated by meta-optimization with a small clean set, which is only 0.2% of the entire training set, while refurbishing the examples probably mislabeled.

For the label refurbishment in (12), *knowledge distillation* [129] adopts the technique of transferring knowledge from one expert model to a target model. The prediction from the expert DNN trained on small clean validation data is used instead of the prediction $\hat{y}$ from the target DNN. *MLC* [130] updates the target model with corrected labels provided by a meta-model trained on clean validation data. The two models are trained concurrently via a bilevel optimization.

Remark: By learning the update rule via meta-learning, the trained network easily adapts to various types of data and label noise. Nevertheless, unbiased clean validation data is essential to minimize the auxiliary objective, although it may not be available in real-world data.

E. Sample Selection

To avoid any false corrections, many recent studies [19], [70], [99], [112], [131]–[137] have adopted sample selection that involves selecting true-labeled examples from a noisy training dataset. In this case, the update equation in (2) is modified to render a DNN more robust for noisy labels. Let $\mathcal{C}_t \subseteq \mathcal{B}_t$ be the identified *clean* examples at time t. Then, the DNN is updated only for the selected clean examples $\mathcal{C}_t$

$$\Theta_{t+1} = \Theta_t - \eta\nabla\left(\frac{1}{|\mathcal{C}_t|}\sum_{(x,\tilde{y})\in\mathcal{C}_t}\ell\big(f(x;\Theta_t),\tilde{y}\big)\right) \tag{14}$$

where the rest mini-batch examples, which are likely to be false-labeled, are excluded to pursue robust learning.

The memorization nature of DNNs has been explored theoretically and empirically to identify clean examples from noisy training data [138]–[140]. Specifically, assuming clusterable data where the clusters are located on the unit Euclidean ball, Li *et al.* [61] proved the distance from the initial weight W_0 to the weight W_t after t iterations

$$\|W_t - W_0\|_F \lesssim \left(\sqrt{K} + (K^2\epsilon_0/\|C\|^2)t\right) \tag{15}$$

where $\|\cdot\|_F$ is the Frobenius norm, K is the number of clusters, and C is the set of cluster centers reaching all input examples within their ϵ_0 neighborhood. Equation (15) demonstrates that the weights of DNNs start to stray far from the initial weights when overfitting to corrupted labels, while they are still in the vicinity of the initial weights at an early stage of training [30], [61]. In the empirical studies [21], [141], the *memorization effect* is also observed since DNNs tend to first learn simple and generalized patterns and then gradually overfit to all noisy patterns. As such, favoring small-loss training examples as the clean ones are commonly employed to design robust training methods [112], [131], [134], [135], [142].

Learning with sample selection is well motivated and works well in general, but this approach suffers from accumulated error caused by incorrect selection, especially when there are many ambiguous classes in training data. Hence, recent approaches often leverage multiple DNNs to cooperate with one another [112] or run multiple training rounds [133]. Moreover, to benefit from even false-labeled examples, loss correction or semisupervised learning have been recently combined with the sample selection strategy [19], [142].

1) Multinetwork Learning: Collaborative learning and co-training are widely used for multinetwork training. Consequently, the sample selection process is guided by the mentor network in the case of collaborative learning or the peer network in the case of co-training.

Technical Detail: Initially, *decouple* [70] proposes the decoupling of when to update from how to update. Hence, two DNNs are maintained simultaneously and updated only the examples selected based on a disagreement between the two DNNs. Next, due to the memorization effect of DNNs, many researchers have adopted another selection criterion, called a *small-loss* trick, which treats a certain number of small-loss training examples as true-labeled examples; many true-labeled examples tend to exhibit smaller losses than false-labeled examples, as shown in Fig. 5(a). In *MentorNet* [131], a pretrained mentor network guides the training of a student network in a collaborative learning manner. Based on the small-loss trick, the mentor network provides the student network with examples whose labels are likely to be correct. *Co-teaching* [112] and *Co-teaching+* [132] also maintain two DNNs, but each DNN selects a certain number of small-loss examples and feeds them to its peer DNN for further training. *Co-teaching+* further employs the disagreement strategy of *decouple* compared with *co-teaching*. In contrast, *JoCoR* [143] reduces the diversity of two networks via co-regularization, making predictions of the two networks closer.

Remark: The co-training methods help reduce the confirmation bias [112], which is a hazard of favoring the examples selected at the beginning of training, while the increase in the number of learnable parameters makes their learning pipeline inefficient. In addition, the small-loss trick does not work well when the loss distribution of true-labeled and false-labeled examples largely overlap, as in the asymmetric noise in Fig. 5(b).

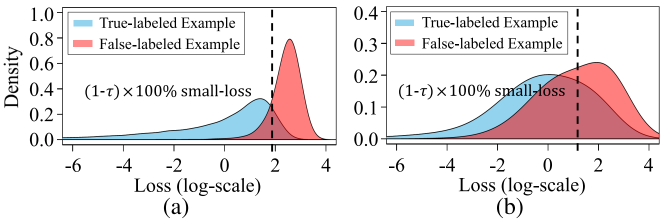

Fig. 5. Loss distribution of training examples at the training accuracy of 50% on noisy CIFAR-100. (This figure is adapted from Song *et al.* [141].) (a) Symmetric noise 40%. (b) Asymmetric noise 40%.

2) Multiround Learning: Without maintaining additional DNNs, multiround learning iteratively refines the selected set of clean examples by repeating the training round. Thus, the selected set keeps improving as the number of rounds increases.

Technical Detail: ITLM [134] iteratively minimizes the trimmed loss by alternating between selecting true-labeled examples at the current moment and retraining the DNN using them. At each training round, only a fraction of small-loss examples obtained in the current round is used to retrain the DNN in the next round. *INCV* [135] randomly divides noisy training data and then employs cross validation to classify true-labeled examples while removing large-loss examples at each training round. Here, *co-teaching* is adopted to train the DNN on the identified examples in the final round of training. Similarly, *O2U-Net* [144] repeats the whole training process with the cyclical learning rate until enough loss statistics of every example are gathered. Next, the DNN is retrained from scratch only for the clean data where false-labeled examples have been detected and removed based on statistics.

A number of variations have been proposed to achieve high performance using iterative refinement only in a single-training round. Beyond the small-loss trick, *iterative detection* [133] detects false-labeled examples by employing the local outlier factor algorithm [145]. With a Siamese network, it gradually pulls away false-labeled examples from true-labeled samples in the deep feature space. *MORPH* [137] introduces the concept of memorized examples which is used to iteratively expand an initial safe set into a maximal safe set via self-transitional learning. *TopoFilter* [146] utilizes the spatial topological pattern of learned representations to detect true-labeled examples, not relying on the prediction of the noisy classifier. *NGC* [147] iteratively constructs the nearest neighbor graph using latent representations and performs geometry-based sample selection by aggregating information from neighborhoods. Soft pesudolabels are assigned to the examples not selected.

Remark: The selected clean set keeps expanded and purified with iterative refinement, mainly through multiround learning. As a side effect, the computational cost for training increases linearly for the number of training rounds.

3) Hybrid Approach: An inherent limitation of sample selection is to discard all the *unselected* training examples, thus resulting in a *partial* exploration of training data.

TABLE II

COMPARISON OF PROPOSED ROBUST DEEP LEARNING METHODS WITH RESPECT TO THE FOLLOWING SIX PROPERTIES: P1—FLEXIBILITY, P2—NO PRETRAINING, P3—FULL EXPLORATION, P4—NO SUPERVISION, P5—HEAVY NOISE, AND P6—COMPLEX NOISE

Category		Method	P1	P2	P3	P4	P5	P6	Implementation
Robust Architecture (§III-A)	Noisy Adaptation Layer	*Webly Learning* [75]	△	×	○	○	×	×	Official (Caffe)[1]
		Noise Model [77]	△	○	○	○	×	×	Unofficial (Keras)[2]
		Dropout Noise Model [78]	△	○	○	○	×	×	Official (MATLAB)[3]
		S-model [79]	△	○	○	○	×	×	Official (Keras)[4]
		C-model [79]	△	○	○	○	×	○	Official (Keras)[4]
		NLNN [76]	△	○	○	○	×	×	Unofficial (Chainer)[5]
	Dedicated Architecture	*Probablistic Noise Model* [16]	×	×	○	×	△	○	Official (Caffe)[6]
		Masking [80]	×	○	○	×	△	○	Official (TensorFlow)[7]
		Contrastive-Additive Noise Network [81]	×	○	○	○	△	○	N/A
		RoG [85]	○	×	○	○	○	△	Official (PyTorch)[8]
Robust Regularization (§III-B)	Explicit Regularization	*Bilevel Learning* [87]	○	○	○	×	△	△	Official (TensorFlow)[9]
		Annotator Confusion [86]	○	×	○	○	△	△	Official (TensorFlow)[10]
		Pre-training [88]	○	×	○	○	△	△	Official (PyTorch)[11]
		PHuber [89]	○	○	○	○	△	△	Unofficial (PyTorch)[12]
		Robust Early-learning [90]	○	○	○	○	△	△	Official (PyTorch)[13]
		ODLN [91]	○	○	○	○	△	△	Official (PyTorch)[14]
	Implicit Regularization	*Adversarial Training* [92]	○	○	○	○	×	△	Unofficial (PyTorch)[15]
		Label Smoothing [93]	○	○	○	○	×	△	Unofficial (PyTorch)[16]
		Mixup [95]	○	○	○	○	×	△	Official (PyTorch)[17]
Robust Loss Function (§III-C)		*Robust MAE* [68]	○	○	○	○	×	×	N/A
		Generalized Cross Entropy [97]	○	○	○	○	×	×	Unofficial (PyTorch)[18]
		Symmetric Cross Entropy [98]	○	○	○	○	×	×	Official (Keras)[19]
		Bi-tempered Loss [104]	○	○	○	○	△	△	Official (TensorFlow)[20]
		Curriculum Learning [99]	○	○	○	×	○	△	N/A
		Active Passive Loss [105]	○	○	○	○	○	△	Official (PyTorch)[21]
Loss Adjustment (§III-D)	Loss Correction	*Backward Correction* [62]	○	○	○	×	×	×	Official (Keras)[22]
		Forward Correction [62]	○	○	○	×	×	×	Official (Keras)[22]
		Gold Loss Correction [107]	○	×	○	×	×	×	Official (PyTorch)[23]
		T-revision [113]	○	×	○	○	×	×	Official (PyTorch)[24]
		Dual T [114]	○	×	○	○	△	×	N/A
	Loss Reweigting	*Importance Reweighting* [108]	○	○	○	○	×	△	Unofficial (PyTorch)[25]
		Active Bias [109]	○	○	○	○	×	△	Unofficial (TensorFlow)[26]
		DualGraph [118]	×	○	○	○	○	△	N/A
	Label Refurbishment	*Bootstrapping* [69]	○	○	○	×	×	△	Unofficial (Keras)[27]
		Dynamic Bootstrapping [110]	○	○	○	○	×	△	Official (PyTorch)[28]
		Self-adaptive Training [119]	○	×	○	○	○	△	Official (PyTorch)[29]
		D2L [111]	○	○	○	○	×	△	Official (Keras)[30]
		AdaCorr [121]	○	○	○	○	○	△	Official (PyTorch)[31]
		SEAL [122]	○	×	○	○	○	○	Official (PyTorch)[32]
	Meta Learning	*L2LWS* [126]	×	○	○	×	△	△	Unofficial (TensorFlow)[33]
		CWS [127]	×	○	○	×	△	△	N/A
		Automatic Reweighting [106]	○	○	○	×	△	△	Official (TensorFlow)[34]
		Meta-weight-net [124]	△	○	○	×	△	△	Official (PyTorch)[35]
		Data Coefficients [128]	○	○	○	×	△	△	Official (TensorFlow)[36]
		Knowledge Distillation [129]	○	×	○	×	△	△	N/A
		MLC [130]	○	○	○	×	△	○	Official (PyTorch)[37]
Sample Selection (§III-E)	Multi-Network Learning	*Decouple* [70]	○	○	×	○	×	△	Official (TensorFlow)[38]
		MentorNet [131]	×	×	×	×	○	△	Official (TensorFlow)[39]
		Co-teaching [112]	○	○	×	×	○	△	Official (PyTorch)[40]
		Co-teaching+ [132]	○	○	×	×	○	△	Official (PyTorch)[41]
		JoCoR [143]	○	○	×	×	○	△	Official (PyTorch)[42]
	Multi-Round Learning	*ITLM* [134]	○	○	×	×	○	△	Official (GluonCV)[43]
		INCV [135]	○	○	×	○	○	△	Official (Keras)[44]
		O2U-Net [144]	○	○	×	×	○	△	Unofficial (PyTorch)[45]
		Iterative Detection [133]	○	○	×	○	○	△	Official (Keras)[46]
		MORPH [137]	○	○	×	○	○	△	N/A
		TopoFilter [146]	○	○	×	○	○	△	Official (PyTorch)[47]
		NGC [147]	○	○	○	○	○	△	N/A
	Hybrid Approach	*SELFIE* [19]	○	○	○	×	○	△	Official (TensorFlow)[48]
		SELF [136]	○	○	○	○	○	△	N/A
		DivideMix [142]	○	○	○	○	○	△	Official (PyTorch)[49]
		RoCL [150]	○	○	○	○	○	△	N/A

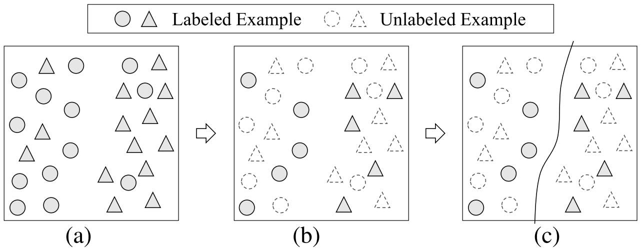

Fig. 6. Procedures for semisupervised learning under label noise. (a) Noisy data. (b) Transformed data. (c) SSL.

To exploit all the noisy examples, researchers have attempted to combine sample selection with other orthogonal ideas.

Technical Detail: The most prominent method in this direction is combining a specific sample selection strategy with a specific semisupervised learning model. As shown in Fig. 6, selected examples are treated as labeled clean data, whereas the remaining examples are treated as unlabeled. Subsequently, semisupervised learning is performed using the transformed data. *SELF* [136] is combined with a semisupervised learning approach to progressively filter out false-labeled examples from noisy data. By maintaining the running average model called the mean-teacher [148] as the backbone, it obtains the self-ensemble predictions of all training examples and then progressively removes examples whose ensemble predictions do not agree with their annotated labels. This method further leverages unsupervised loss from the examples not included in the selected clean set. *DivideMix* [142] uses two-component and 1-D Gaussian mixture models to transform noisy data into labeled (clean) and unlabeled (noisy) sets. Then, it applies a semisupervised technique *MixMatch* [149]. Recently, *RoCL* [150] employs two-phase learning strategies: supervised training on selected clean examples and then semisupervised learning on relabeled noisy examples with self-supervision. For selection and relabeling, it computes the exponential moving average of the loss over training iterations.

Meanwhile, *SELFIE* [19] is a hybrid approach of sample selection and loss correction. The loss of refurbishable examples is corrected (i.e., loss correction) and then used together with that of small-loss examples (i.e., sample selection). Consequently, more training examples are considered for updating the DNN. The *CL* [99] is combined with the robust loss function approach and used to extract the true-labeled examples from noisy data.

Remark: Noise robustness is significantly improved by combining with other techniques. However, the hyperparameters introduced by these techniques render a DNN more susceptible to changes in data and noise types, and an increase in computational cost is inevitable.

IV. Methodological Comparison

In this section, we compare the 62 deep learning methods for overcoming noisy labels introduced in Section III with respect to the following *six* properties. When selecting the properties, we refer to the properties that are typically used to compare the performance of robust deep learning methods [19], [112]. To the best of our knowledge, this survey is the first to provide a systematic comparison of robust training methods. This comprehensive comparison will provide useful insights that can enlighten new future directions.

1) *(P1) Flexibility:* With the rapid evolution of deep learning research, a number of new network architectures are constantly emerging and becoming available. Hence, the ability to support any type of architecture is important. “Flexibility” ensures that the proposed method can quickly adapt to the state-of-the-art architecture.
2) *(P2) No Pretraining:* A typical approach to improve noise robustness is to use a pretrained network; however, this incurs an additional computational cost to the learning process. “No Pretraining” ensures that the proposed method can be trained from scratch without any pretraining.

[1] https://github.com/endernewton/webly-supervised
[2] https://github.com/delchiaro/training-cnn-noisy-labels-keras
[3] https://github.com/ijindal/Noisy_Dropout_regularization
[4] https://github.com/udibr/noisy_labels
[5] https://github.com/Ryo-Ito/Noisy-Labels-Neural-Network
[6] https://github.com/Cysu/noisy_label
[7] https://github.com/bhanML/Masking
[8] https://github.com/pokaxpoka/RoGNoisyLabel
[9] https://github.com/sjenni/DeepBilevel
[10] https://rt416.github.io/pdf/trace_codes.pdf
[11] github.com/hendrycks/pre-training
[12] https://github.com/dmizr/phuber
[13] https://github.com/xiaoboxia/CDR
[14] https://github.com/hongxin001/ODNL?ref=pythonrepo.com
[15] https://https://github.com/sarathknv/adversarial-examples-pytorch
[16] https://github.com/CoinCheung/pytorch-loss
[17] https://github.com/facebookresearch/mixup-cifar10
[18] https://github.com/AlanChou/Truncated-Loss
[19] https://github.com/YisenWang/symmetric_cross_entropy
[20] https://github.com/google/bi-tempered-loss
[21] https://github.com/HanxunH/Active-Passive-Losses
[22] https://github.com/giorgiop/loss-correction
[23] https://github.com/mmazeika/glc
[24] https://github.com/xiaoboxia/T-Revision
[25] https://github.com/xiaoboxia/Classification-with-noisy-labels
[26] https://github.com/songhwanjun/ActiveBias
[27] https://github.com/dr-darryl-wright/Noisy-Labels-with-Bootstrapping
[28] https://github.com/PaulAlbert31/LabelNoiseCorrection
[29] https://github.com/LayneH/self-adaptive-training
[30] https://github.com/xingjunm/dimensionality-driven-learning
[31] https://github.com/pingqingsheng/LRT
[32] https://github.com/chenpf1025/IDN
[33] https://github.com/krayush07/learn-by-weak-supervision
[34] https://github.com/uber-research/learning-to-reweight-examples
[35] https://github.com/xjtushujun/meta-weight-net
[36] https://github.com/google-research/google-research/tree/master/ieg
[37] https://aka.ms/MLC
[38] https://github.com/emalach/UpdateByDisagreement
[39] https://github.com/google/mentornet
[40] https://github.com/bhanML/Co-teaching
[41] https://github.com/bhanML/coteaching_plus
[42] https://github.com/hongxin001/JoCoR
[43] https://github.com/yanyao-shen/ITLM-simplecode
[44] https://github.com/chenpf1025/noisy_label_understanding_utilizing
[45] https://github.com/hjimce/O2U-Net
[46] https://github.com/YisenWang/Iterative_learning
[47] https://github.com/pxiangwu/TopoFilter
[48] https://github.com/kaist-dmlab/SELFIE
[49] https://github.com/LiJunnan1992/DivideMix

TABLE III

COMPARISON OF ROBUST DEEP LEARNING CATEGORIES FOR OVERCOMING NOISY LABELS

Category		**P1** Flexibility	**P2** No Pre-train	**P3** Full Exploration	**P4** No Supervision	**P5** Heavy Noise	**P6** Complex Noise
Robust Architecture (§III-A)	Noise Adaptation Layer	△	◯	◯	◯	✕	✕
	Dedicated Architecture	✕	△	◯	△	△	◯
Robust Regularization (§III-B)	Implicit Regularization	◯	◯	◯	◯	△	△
	Explicit Regularization	◯	◯	◯	◯	✕	△
Robust Loss Function (§III-C)		◯	◯	◯	◯	✕	✕
Loss Adjustment (§III-D)	Loss Correction	◯	✕	◯	✕	✕	✕
	Loss Reweighting	◯	◯	◯	◯	✕	△
	Label Refurbishment	◯	◯	◯	◯	△	△
	Meta Learning	◯	◯	◯	✕	△	△
Sample Selection (§III-E)	Multi-Network Learning	◯	◯	✕	✕	◯	△
	Multi-Round Learning	◯	◯	✕	◯	◯	△
	Hybrid Approach	◯	◯	◯	◯	◯	△

3) *(P3) Full Exploration:* Excluding unreliable examples from the update is an effective method for robust deep learning; however, it eliminates hard but useful training examples as well. "Full exploration" ensures that the proposed methods can use *all* training examples without severe overfitting to false-labeled examples by adjusting their training losses or applying semisupervised learning.
4) *(P4) No Supervision:* Learning with supervision, such as a clean validation set or a known noise rate, is often impractical because they are difficult to obtain. Hence, such supervision had better be avoided to increase practicality in real-world scenarios. "No supervision" ensures that the proposed methods can be trained without any supervision.
5) *(P5) Heavy Noise:* In real-world noisy data, the noise rate can vary from light to heavy. Hence, learning methods should achieve consistent noise robustness with respect to the noise rate. "Heavy noise" ensures that the proposed methods can combat even the heavy noise.
6) *(P6) Complex Noise:* The type of label noise significantly affects the performance of a learning method. To manage real-world noisy data, diverse types of label noise should be considered when designing a robust training method. "Complex noise" ensures that the proposed method can combat even the complex label noise.

Table II shows a comparison of all robust deep learning methods, which are grouped according to the most appropriate category. In the first row, the aforementioned six properties are labeled as P1–P6, and the availability of open-source implementation is added in the last column. For each property, we assign "◯" if it is completely supported, "✕" if it is not supported, and "△" if it is supported but not completely. More specifically, "△" is assigned to P1 if the method can be flexible but requires additional effort, to P5 if the method can combat only moderate label noise, and to P6 if the method does not make a strict assumption about the noise type but without explicitly modeling instance-dependent noise. Thus, for P6, the method marked with "✕" only deals with the instance-independent noise, while the method marked with "◯" deals with both instance-independent and -dependent noises. The remaining properties (i.e., P2–P4) are only assigned "◯" or "✕." Regarding the implementation, we assign "N/A" if a publicly available source code is not available.

No existing method supports all the properties. Each method achieves noise robustness by supporting a different combination of the properties. The supported properties are similar among the methods of the same (sub)category because those methods share the same methodological philosophy; however, they differ significantly depending on the (sub)category. Therefore, we investigate the properties generally supported in each (sub)category and summarize them in Table III. Here, the property of a (sub)category is marked as the majority of the belonging methods. If no clear trend is observed among those methods, then the property is marked "△."

V. NOISE RATE ESTIMATION

The estimation of a noise rate is an imperative part of utilizing robust methods for better practical use, especially with the approaches belonging to the loss adjustment and sample selection. The estimated noise rate is widely used to reweight examples for a robust classifier [97], [114], [117] or to determine how many examples should be selected as clean ones [19], [112], [135]. However, detailed analysis has yet to be performed properly, though many robust approaches highly rely on the accuracy of noise rate estimation. The noise rate can be estimated by exploiting the inferred noise transition matrix [113], [114], [151], the Gaussian mixture model [110], [137], [152], or the cross validation [19], [135].

A. Noise Transition Matrix

The noise transition matrix has been used to build a statistically consistent robust classifier because it represents the class posterior probabilities for noisy and clean data, as in (3). The first method to estimate the noise rate is exploiting this noise transition matrix, which can be inferred or trained accurately by using perfectly clean examples, i.e., *anchor points* [117], [153]; an example x with its label i is defined as an anchor point if $p(y = i|x) = 1$ and $p(y = k|x) = 0$ for $k \neq i$. Thus, let $\mathcal{A}_i$ be the set of anchor points with label i, then the

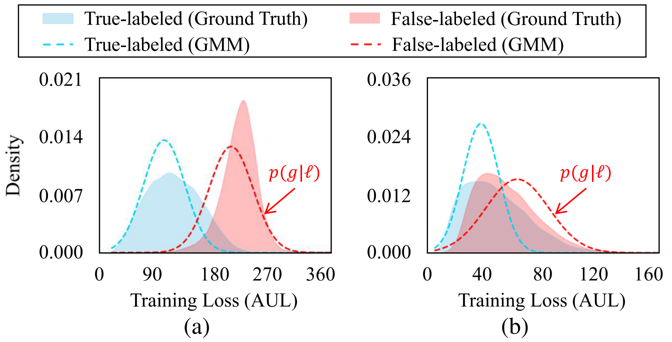

Fig. 7. Training loss distributions of true-labeled and false-labeled examples using the ground-truth label and the GMM on CIFAR-100 data with two synthetic noises of 40%. (a) Symmetric noise. (b) Asymmetric noise.

element of the noise transition matrix T_{ij} is estimated by

$$\begin{aligned}\hat{T}_{ij} &= \frac{1}{|\mathcal{A}_i|}\sum_{x\in\mathcal{A}_i}\sum_{k=1}^{c} p(\tilde{y}=j|y=k)p(y=k|x)\\ &= \frac{1}{|\mathcal{A}_i|}\sum_{x\in\mathcal{A}_i} p(\tilde{y}=j|x;\Theta)\end{aligned} \tag{16}$$

where $p(\tilde{y}=j|x;\Theta)$ is the noisy class posterior probability of the classifier trained on noisy training data for the anchor point x (see the detailed proof in [107], [113], and [114]). Next, based on the inferred noise transition matrix, the noise rate of a balanced training dataset is obtained by averaging the label transition probabilities between classes

$$\hat{\tau} = \frac{1}{c}\sum_{i=1}^{c}\sum_{j\neq i}^{c} p(\tilde{y}=j|y=i) = \frac{1}{c}\sum_{i=1}^{c}\sum_{j\neq i}^{c}\hat{T}_{ij}. \tag{17}$$

However, since the anchor points are typically unknown in real-world data, they are identified from noisy training data by either theoretical derivations [117] or heuristics [62]. In addition, there have been recent efforts to learn the noise transition matrix without anchor points. *T-revision* [113] initializes a transition matrix by exploiting the examples with high noisy class posterior probabilities and then refines the matrix by adding a slack variable. *Dual-T* [114] introduces an intermediate class that factorizes the transition matrix into two easy-to-estimate matrices for better accuracy. *VolMinNet* [151] realizes an end-to-end framework and relaxes the need for anchor points under the sufficiently scattered assumption.

B. Gaussian Mixture Model

The second method is exploiting a 1-D and two-component GMM to model the loss distribution of true-labeled and false-labeled examples [110], [152]. As shown in Fig. 7, since the loss distribution tends to be *bimodal*, the two Gaussian components are fit to the training loss by using the EM algorithm; the probability of an example being a false-labeled one is obtained through its posterior probability. Hence, the noise rate is estimated at each epoch t by computing the expectation of the posterior probability for all training examples

$$\hat{\tau} = \mathbb{E}_{(x,\tilde{y})\in\tilde{\mathcal{D}}}\Big[p\big(g|\ell\big(f(x;\Theta_t),\tilde{y}\big)\big)\Big] \tag{18}$$

where g is the Gaussian component with a larger loss. However, Pleiss *et al.* [152] recently pointed out that the training loss becomes less separable by the GMM as the training progresses, and thus, proposed the *area under the loss* (AUL) curve, which is the sum of the example's training losses obtained from all previous training epochs. Even after the loss signal decays in later epochs, the distributions remain separable. Therefore, the noise rate is finally estimated by

$$\hat{\tau} = \mathbb{E}_{(x,\tilde{y})\in\tilde{\mathcal{D}}}\Big[p\big(g|\mathrm{AUL}_t(x,\tilde{y})\big)\Big]$$
$$\text{where } \mathrm{AUL}_t(x,\tilde{y}) = \sum_{i=1}^{t}\ell\big(f(x;\Theta_i),\tilde{y}\big). \tag{19}$$

C. Cross Validation

The third method is estimating the noise rate by applying cross validation, which typically requires clean validation data [19], [112], [132]. However, such clean validation data are hard to acquire in real-world applications. Thus, Chen *et al.* [135] leveraged two randomly divided noisy training datasets for cross validation. Under the assumption that the two datasets share exactly the same noise transition matrix, the noise rate quantifies the test accuracy of DNNs that are, respectively, trained and tested on the two divided sets

$$\mathrm{TestAccuracy} = \begin{cases}(1-\hat{\tau})^2 + \hat{\tau}^2/(c-1), & \text{if symmetric}\\ (1-\hat{\tau})^2 + \hat{\tau}^2, & \text{if asymmetric.}\end{cases} \tag{20}$$

Therefore, the noise rate is estimated from the test accuracy obtained by cross validation.

VI. Experimental Design

This section describes the typically used experimental design for comparing robust training methods in the presence of label noise. We introduce publicly available image datasets and then describe widely used evaluation metrics.

A. Publicly Available Datasets

To validate the robustness of the proposed algorithms, an image classification task was widely conducted on numerous image benchmark datasets. Table IV summarizes popularly used public benchmark datasets, which are classified into two categories: 1) a "clean dataset" that consists of mostly true-labeled examples annotated by human experts and 2) a "real-world noisy dataset" that comprises real-world noisy examples with varying numbers of false labels.

1) Clean Datasets: According to the literature [19], [133], [142], *seven* clean datasets are widely used: MNIST[50], classification of handwritten digits [154]; fashion-MNIST[51], classification of various clothing [155]; CIFAR-10[52], and CIFAR-100,[52] classification of a subset of 80 million categorical images [156]; SVHN,[53] classification of house numbers in Google Street view images [157]; ImageNet[54], and

[50] http://yann.lecun.com/exdb/mnist

[51] https://github.com/zalandoresearch/fashion-mnist

[52] https://www.cs.toronto.edu/~kriz/cifar.html

[53] http://ufldl.stanford.edu/housenumbers

[54] http://www.image-net.org

TABLE IV

SUMMARY OF PUBLICLY AVAILABLE DATASETS USED FOR STUDYING LABEL NOISE

Dataset		# Training	# Validation	# Testing	# Classes	Noise Rate (%)
Clean Data	MNIST [154][50]	60K	N/A	10K	10	≈ 0.0
	Fashion-MNIST [155][51]	60K	N/A	10K	10	≈ 0.0
	CIFAR-10 [156][52]	50K	N/A	10K	10	≈ 0.0
	CIFAR-100 [156][52]	50K	N/A	10K	100	≈ 0.0
	SVHN [157][53]	73K	N/A	26K	10	≈ 0.0
	Tiny-ImageNet [158][55]	100K	10K	10K	200	≈ 0.0
	ImageNet [1][54]	1.3M	50K	50K	1000	≈ 0.0
Real-world Noisy Data	ANIMAL-10N [19][56]	50K	N/A	5K	10	≈ 8.0
	CIFAR-10N [159][57]	50K	N/A	10K	10	$\approx 9.0/18.0/40.2$
	CIFAR-100N [159][57]	50K	N/A	10K	100	$\approx 25.6/40.2$
	Food-101N [18][58]	310K	5K	25K	101	≈ 18.4
	Clothing1M [16][59]	1M	14K	10K	14	≈ 38.5
	WebVision [17][60]	2.4M	50K	50K	1000	≈ 20.0

tiny-ImageNet,[55] image database organized according to the WordNet hierarchy and its small subset [1], [158]. Because the labels in these datasets are almost all true-labeled, their labels in the training data should be artificially corrupted for the evaluation of synthetic noises, namely *symmetric* noise and *asymmetric* noise.

2) *Real-World Noisy Datasets:* Unlike the clean datasets, real-world noisy datasets inherently contain many mislabeled examples annotated by nonexperts. According to the literature [16]–[19], *six* real-world noisy datasets are widely used: ANIMAL-10N,[56] real-world noisy data of human-labeled online images for 10 confusing animals [19]; CIFAR-10N[57] and CIFAR-100N,[57] variations of CIFAR-10 and CIFAR-100 with human-annotated real-world noisy labels collected from Amazon's Mechanical Turk [159]. They provide human labels with different noise rates, as shown in Table IV; Food-101N,[58] real-world noisy data of crawled food images annotated by their search keywords in the Food-101 taxonomy [18], [160]; Clothing1M,[59] real-world noisy data of large-scale crawled clothing images from several online shopping websites [16]; WebVision,[60] real-world noisy data of large-scale web images crawled from Flickr and Google Images search [17]. To support sophisticated evaluation, most real-world noisy datasets contain their own clean validation set and provide the estimated noise rate of their training set.

B. Evaluation Metrics

A typical metric to assess the robustness of a particular method is the prediction accuracy for unbiased and clean examples that are not used in training. The prediction accuracy degrades significantly if the DNN overfits to false-labeled examples [22]. Hence, *test accuracy* has generally been adopted for evaluation [13]. For a test set $\mathcal{T} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{T}|}$, let $\hat{y}_i$ be the predicted label of the ith example in $\mathcal{T}$. Subsequently, the test accuracy is formalized by

$$\text{Test Accuracy} = \frac{|\{(x_i, y_i) \in \mathcal{T} : \hat{y}_i = y_i\}|}{|\mathcal{T}|}. \tag{21}$$

If the test data are not available, *validation accuracy* can be used by replacing $\mathcal{T}$ in (21) with validation data $\mathcal{V} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{V}|}$ as an alternative

$$\text{Validation Accuracy} = \frac{|\{(x_i, y_i) \in \mathcal{V} : \hat{y}_i = y_i\}|}{|\mathcal{V}|}. \tag{22}$$

Furthermore, if the specified method belongs to the "sample selection" category, *label precision* and *label recall* [112], [135] can be used as the metrics

$$\begin{aligned}\text{Label Precision} &= \frac{|\{(x_i, \tilde{y}_i) \in \mathcal{S}_t : \tilde{y}_i = y_i\}|}{|\mathcal{S}_t|}\\ \text{Label Recall} &= \frac{|\{(x_i, \tilde{y}_i) \in \mathcal{S}_t : \tilde{y}_i = y_i\}|}{|\{(x_i, \tilde{y}_i) \in \mathcal{B}_t : \tilde{y}_i = y_i\}|}\end{aligned} \tag{23}$$

where $\mathcal{S}_t$ is the set of selected clean examples in a mini-batch $\mathcal{B}_t$. The two metrics are performance indicators for the examples selected from the mini-batch as true-labeled ones [112].

Meanwhile, if the specified method belongs to the "label refurbishment" category, *correction error* [19] can be used as an indicator of how many examples are incorrectly refurbished

$$\text{Correction Error} = \frac{\left|\left\{x_i \in \mathcal{R} : \text{argmax}\left(y_i^{\text{refurb}}\right) \neq y_i\right\}\right|}{|\mathcal{R}|} \tag{24}$$

where $\mathcal{R}$ is the set of examples whose labels are refurbished by (12) and y_i^{refurb} is the refurbished label of the ith examples in $\mathcal{R}$.

VII. FUTURE RESEARCH DIRECTIONS

With recent efforts in the machine learning community, the robustness of DNNs becomes evolved in several directions. Thus, the existing approaches covered in our survey face a variety of future challenges. This section provides a discussion for future research that can facilitate and envision the development of deep learning in the label noise area.

[55] https://www.kaggle.com/c/tiny-imagenet

[56] https://dm.kaist.ac.kr/datasets/animal-10n

[57] http://noisylabels.com/

[58] https://kuanghuei.github.io/Food-101N

[59] https://www.floydhub.com/lukasmyth/datasets/clothing1m

[60] https://data.vision.ee.ethz.ch/cvl/webvision/download.html

A. Instance-Dependent Label Noise

Existing theoretical and empirical studies for *robust loss function* and *loss correction* are largely built upon the instance-independent noise assumption that the label noise is independent of input features [76], [77], [113], [114]. However, this assumption may not be a good approximation of the real-world label noise. In particular, Chen *et al.* [122] conducted a theoretical hypothesis testing[61] using a popular real-world dataset, Clothing1M, and proved that its label noise is statistically different from the instance-independent noise. This testing confirms that the label noise should depend on the instance.

Conversely, most methods for the other direction (especially, *sample selection*) work well even under the instance-dependent label noise in general since they do not rely on the assumption. Nevertheless, Song *et al.* [141] pointed out that their performance could considerably worsen in the instance-dependent (or real-world) noise compared with symmetric noise due to the confusion between true-labeled and false-labeled examples. The loss distribution of true-labeled examples heavily overlaps that of false-labeled samples in the asymmetric noise, which is similar to the real-world noise, in Fig. 5(b). Thus, identifying clean examples becomes more challenging when dealing with the instance-dependent label noise.

Beyond the instance-independent label noise, there have been a few recent studies for the instance-dependent label noise. Mostly, they only focus on a binary classification task [66], [161] or a restricted small-scale machine learning model, such as logistic regression [63]. Therefore, learning with the instance-dependent label noise is an important topic that deserves more research attention.

B. Multilabel Data With Label Noise

Most of the existing methods are applicable only for a *single-label* multiclass classification problem, where each data example is assumed to have only one true label. However, in the case of *multilabel* learning, each data example can be associated with a set of multiple true class labels. In music categorization, each music can belong to multiple categories [162]. In semantic scene classification, each scene may belong to multiple scene classes [163]. Thus, contrary to the single-label setup, the multilabel classifier aims to predict a set of target objects simultaneously. In this setup, a multilabel dataset of millions of examples is reported to contain over 26.6% false-positive labels and a significant number of omitted labels [164].

Even worse, the difference in occurrence between classes makes this problem more challenging; some minor class labels occur less in training data than other major class labels. Considering such aspects that can arise in multilabel classification, the simple extension of existing methods may not learn the proper correlations among multiple labels. Therefore, learning from noisy labels with multilabel data is another important topic for future research. We refer the readers to a recent study [165] that discusses the evaluation of multilabel classifiers trained with noisy labels.

C. Class Imbalance Data With Label Noise

The *class imbalance* in training data is commonly observed, where a few classes account for most of the data. Especially when working with large data in many real-world applications, this problem becomes more severe and is often associated with the problem of noisy labels [166]. Nevertheless, to ease the label noise problem, it is commonly assumed that training examples are equally distributed over all class labels in the training data. This assumption is quite strong when collecting large-scale data, and thus, we need to consider a more realistic scenario in which the two problems coexist.

Most of the existing robust methods may not work well with the class imbalance, especially when they rely on the learning dynamics of DNNs, e.g., the small-loss trick or memorization effect. Under the existence of the class imbalance, the training model converges to major classes faster than minor classes such that most examples in the major class exhibit small losses (i.e., early memorization). That is, there is a risk of discarding most examples in the minor class. Furthermore, in terms of example importance, high-loss examples are commonly favored for the class imbalance problem [124], while small-loss examples are favored for the label noise problem. This conceptual contradiction hinders the applicability of the existing methods that neglect the class imbalance. Therefore, these two problems should be considered simultaneously to deal with more general situations.

D. Robust and Fair Training

Machine learning classifiers can perpetuate and amplify the existing systemic injustices in society [167]. Hence, fairness is becoming another important topic. Traditionally, robust training and fair training have been studied by separate communities; robust training with noisy labels has mostly focused on combating label noise without regarding data bias [13], [30], whereas fair training has focused on dealing with data bias, not necessarily noise [167], [168]. However, noisy labels and data bias, in fact, coexist in real-world data. Satisfying both robustness and fairness is more realistic but challenging because the bias in data is pertinent to label noise.

In general, many fairness criteria are group-based, where a target metric is equalized or enforced over subpopulations in the data, also known as *protected groups*, such as race or gender [167]. Accordingly, the goal of fair training is building a model that satisfies such fairness criteria for the *true* protected groups. However, if the *noisy* protection group is involved, such fairness criteria cannot be directly applied. Recently, mostly after 2020, a few pioneering studies have emerged to consider both robustness and fairness objectives at the same time under the binary classification setting [169], [170]. Therefore, more research attention is needed for the convergence of robust training and fair training.

E. Connection With Input Perturbation

There has been a lot of research on the robustness of deep learning under input perturbation, mainly in the field

[61] In Clothing1M, the result showed that the instance-independent noise happens with probability lower than 10^{-21250}, which is statistically impossible.

of adversarial training where the input feature is maliciously perturbed to distort the output of the DNN [34], [36]. Although learning with noisy labels and learning with noisy inputs have been regarded as separate research fields, their goals are similar in that they learn noise-robust representations from noisy data. Based on this common point of view, a few recent studies have investigated the interaction of adversarial training with noisy labels [171]–[173].

Interestingly, it was turned out that adversarial training makes DNNs robust to label noise [171]. Based on this finding, Damodaran *et al.* [172] proposed a new regularization term, called Wasserstein adversarial regularization, to address the problem of learning with noisy labels. Zhu *et al.* [173] proposed to use the number of projected gradient descent steps as a new criterion for sample selection such that clean examples are filtered out from noisy data. These approaches are regarded as a new perspective on label noise compared with traditional work. Therefore, understanding the connection between input perturbation and label noise could be another future topic for better representation learning toward robustness.

F. Efficient Learning Pipeline

The efficiency of the learning pipeline is another important aspect to design deep learning approaches. However, for robust deep learning, most studies have neglected the efficiency of the algorithm because their main goal is to improve the robustness to label noise. For example, maintaining multiple DNNs or training a DNN in multiple rounds is frequently used, but these approaches significantly degrade the efficiency of the learning pipeline. On the other hand, the need for more efficient algorithms is increasing owing to the rapid increase in the amount of available data [174].

According to our literature survey, most work did not even report the efficiency (or time complexity) of their approaches. However, it is evident that saving the training time is helpful under the restricted budget for computation. Therefore, enhancing the efficiency will significantly increase the usability of robust deep learning in the big data era.

VIII. Conclusion

DNNs easily overfit false labels owing to their high capacity in totally memorizing all noisy training samples. This overfitting issue still remains even with various conventional regularization techniques, such as dropout and batch normalization, thereby significantly decreasing their generalization performance. Even worse, in real-world applications, the difficulty in labeling renders the overfitting issue more severe. Therefore, learning from noisy labels has recently become one of the most active research topics.

In this survey, we presented a comprehensive understanding of modern deep learning methods to address the negative consequences of learning from noisy labels. All the methods were grouped into five categories according to their underlying strategies and described along with their methodological weaknesses. Furthermore, a systematic comparison was conducted using six popular properties used for evaluation in the recent literature. According to the comparison results, there is no ideal method that supports all the required properties; the supported properties varied depending on the category to which each method belonged. Several experimental guidelines were also discussed, including noise rate estimation, publicly available datasets, and evaluation metrics. Finally, we provided insights and directions for future research in this domain.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. CVPR*, 2016, pp. 779–788.

[3] W. Zhang, T. Du, and J. Wang, "Deep learning over multi-field categorical data," in *Proc. ECIR*, 2016, pp. 45–57.

[4] L. Pang, Y. Lan, J. Guo, J. Xu, J. Xu, and X. Cheng, "DeepRank: A new deep architecture for relevance ranking in information retrieval," in *Proc. CIKM*, 2017, pp. 257–266.

[5] K. D. Onal *et al.*, "Neural information retrieval: At the end of the early years," *Inf. Retr. J.*, vol. 21, nos. 2–3, pp. 111–182, 2018.

[6] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proc. ACL*, 2018, pp. 328–339.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. ACL*, 2019, pp. 4171–4186.

[8] A. Severyn and A. Moschitti, "Twitter sentiment analysis with deep convolutional neural networks," in *Proc. ACL*, 2015, pp. 959–962.

[9] G. Paolacci, J. Chandler, and P. G. Ipeirotis, "Running experiments on Amazon mechanical Turk," *Judgment Decis. Making*, vol. 5, no. 5, pp. 411–419, 2010.

[10] V. Cothey, "Web-crawling reliability," *J. Amer. Soc. Inf. Sci. Technol.*, vol. 55, no. 14, pp. 1228–1238, 2004.

[11] W. Mason and S. Suri, "Conducting behavioral research on Amazon's mechanical Turk," *Behav. Res. Methods*, vol. 44, no. 1, pp. 1–23, 2012.

[12] C. Scott, G. Blanchard, and G. Handy, "Classification with asymmetric label noise: Consistency and maximal denoising," in *Proc. COLT*, 2013, pp. 489–511.

[13] B. Frenay and M. Verleysen, "Classification in the presence of label noise: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 5, pp. 845–869, May 2014.

[14] R. V. Lloyd *et al.*, "Observer variation in the diagnosis of follicular variant of papillary thyroid carcinoma," *The Amer. J. Surgical Pathol.*, vol. 28, no. 10, pp. 1336–1340, 2004.

[15] H. Xiao, H. Xiao, and C. Eckert, "Adversarial label flips attack on support vector machines," in *Proc. ECAI*, 2012, pp. 870–875.

[16] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, "Learning from massive noisy labeled data for image classification," in *Proc. CVPR*, 2015, pp. 2691–2699.

[17] W. Li, L. Wang, W. Li, E. Agustsson, and L. Van Gool, "Web vision database: Visual learning and understanding from web data," 2017, *arXiv:1708.02862*.

[18] K.-H. Lee, X. He, L. Zhang, and L. Yang, "CleanNet: Transfer learning for scalable image classifier training with label noise," in *Proc. CVPR*, 2018, pp. 5447–5456.

[19] H. Song, M. Kim, and J.-G. Lee, "SELFIE: Refurbishing unclean samples for robust deep learning," in *Proc. ICML*, 2019, pp. 5907–5915.

[20] J. Krause *et al.*, "The unreasonable effectiveness of noisy data for fine-grained recognition," in *Proc. ECCV*, 2016, pp. 301–320.

[21] D. Arpit *et al.*, "A closer look at memorization in deep networks," in *Proc. ICML*, 2017, pp. 233–242.

[22] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *Proc. ICLR*, 2017.

[23] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, p. 60, Dec. 2019.

[24] A. Krogh and J. A. Hertz, "A simple weight decay can improve generalization," in *Proc. NeurIPS*, 1992, pp. 950–957.

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "DropOut: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[26] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[27] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study," *Artif. Intell. Rev.*, vol. 22, no. 3, pp. 177–210, Nov. 2004.

[28] J. Zhang, X. Wu, and V. S. Sheng, "Learning from crowdsourced labeled data: A survey," *Artif. Intell. Rev.*, vol. 46, no. 4, pp. 543–576, Dec. 2016.

[29] N. Nigam, T. Dutta, and H. P. Gupta, "Impact of noisy labels in learning techniques: A survey," in *Proc. ICDIS*, 2020, pp. 403–411.

[30] B. Han *et al.*, "A survey of label-noise representation learning: Past, present and future," 2020, *arXiv:2011.04406*.

[31] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14410–14430, 2018.

[32] J. Yoon, J. Jordon, and M. Schaar, "GAIN: Missing data imputation using generative adversarial nets," in *Proc. ICML*, 2018, pp. 5689–5698.

[33] A. Fawzi, S.-M. Moosavi-Dezfooli, and P. Frossard, "Robustness of classifiers: From adversarial to random noise," in *Proc. NeurIPS*, 2016, pp. 1632–1640.

[34] E. Dohmatob, "Generalized no free lunch theorem for adversarial robustness," in *Proc. ICML*, 2019, pp. 1646–1654.

[35] J. Gilmer, N. Ford, N. Carlini, and E. Cubuk, "Adversarial examples are a natural consequence of test error in noise," in *Proc. ICML*, 2019, pp. 2280–2289.

[36] S. Mahloujifar, D. I. Diochnos, and M. Mahmoody, "The curse of concentration in robust learning: Evasion and poisoning attacks from concentration of measure," in *Proc. AAAI*, 2019, vol. 33, no. 1, pp. 4536–4543.

[37] D. B. Rubin, "Inference and missing data," *Biometrika*, vol. 63, no. 3, pp. 581–592, 1976.

[38] C. M. Bishop, *Pattern Recognition Machine Learning*. Springer, 2006.

[39] N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari, "Learning with noisy labels," in *Proc. NeurIPS*, 2013, pp. 1196–1204.

[40] J. Goldberger and E. Ben-Reuven, "Training deep neural-networks using a noise adaptation layer," in *Proc. ICLR*, 2017, pp. 1–9.

[41] P. Sastry and N. Manwani, "Robust learning of classifiers in the presence of label noise," in *Proc. Pattern Recognit. Big Data*, 2017, pp. 167–197.

[42] V. Wheway, "Using boosting to detect noisy data," in *Proc. PRICAI*, 2000, pp. 123–130.

[43] B. Sluban, D. Gamberger, and N. Lavrač, "Ensemble-based noise detection: Noise ranking and visual performance evaluation," *Data Mining Knowl. Discovery*, vol. 28, no. 2, pp. 265–303, 2012.

[44] S. J. Delany, N. Segata, and B. Mac Namee, "Profiling instances in noise reduction," *Knowl.-Based Syst.*, vol. 31, pp. 28–40, Jul. 2012.

[45] D. Gamberger, N. Lavrac, and S. Dzeroski, "Noise detection and elimination in data preprocessing: Experiments in medical domains," *Appl. Artif. Intell.*, vol. 14, no. 2, pp. 205–223, Nov. 2000.

[46] J. Thongkam, G. Xu, Y. Zhang, and F. Huang, "Support vector machine for outlier detection in breast cancer survivability prediction," in *Proc. APWeb*, 2008, pp. 99–109.

[47] V. Mnih and G. E. Hinton, "Learning to label aerial images from noisy data," in *Proc. ICML*, 2012, pp. 567–574.

[48] N. Manwani and P. Sastry, "Noise tolerance under risk minimization," *IEEE Trans. Cybern.*, vol. 43, no. 3, pp. 1146–1151, Jun. 2013.

[49] A. Ghosh, N. Manwani, and P. S. Sastry, "Making risk minimization tolerant to label noise," *Neurocomputing*, vol. 160, pp. 93–107, Jul. 2015.

[50] B. Van Rooyen, A. Menon, and R. C. Williamson, "Learning with symmetric label noise: The importance of being unhinged," in *Proc. NeurIPS*, 2015, pp. 10–18.

[51] G. Patrini, F. Nielsen, R. Nock, and M. Carioni, "Loss factorization, weakly supervised learning and label noise robustness," in *Proc. ICML*, 2016, pp. 708–717.

[52] R. Xu and D. C. Wunsch, "Survey of clustering algorithms," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, Jun. 2005.

[53] U. Rebbapragada and C. E. Brodley, "Class noise mitigation through instance weighting," in *Proc. ECML*, 2007, pp. 708–715.

[54] T. Liu, K. Wang, B. Chang, and Z. Sui, "A soft-label method for noise-tolerant distantly supervised relation extraction," in *Proc. EMNLP*, 2017, pp. 1790–1795.

[55] F. O. Kaster, B. H. Menze, M.-A. Weber, and F. A. Hamprecht, "Comparative validation of graphical models for learning tumor segmentations from noisy manual annotations," in *Proc. MICCAI*, 2010, pp. 74–85.

[56] A. Ganapathiraju and J. Picone, "Support vector machines for automatic data cleanup," in *Proc. ICSLP*, 2000, pp. 210–213.

[57] B. Biggio, B. Nelson, and P. Laskov, "Support vector machines under adversarial label noise," in *Proc. ACML*, 2011, pp. 97–112.

[58] C. J. Mantas and J. Abellán, "Credal-C4.5: Decision tree based on imprecise probabilities to classify noisy data," *Expert Syst. Appl.*, vol. 41, no. 10, pp. 4625–4637, Aug. 2014.

[59] A. Ghosh, N. Manwani, and P. Sastry, "On the robustness of decision tree learning under label noise," in *Proc. PAKDD*, 2017, pp. 685–697.

[60] S. Liu, J. Niles-Weed, N. Razavian, and C. Fernandez-Granda, "Early-learning regularization prevents memorization of noisy labels," in *Proc. NeurIPS*, vol. 33, 2020, pp. 20331–20342.

[61] M. Li, M. Soltanolkotabi, and S. Oymak, "Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks," in *Proc. AISTATS*, 2020, pp. 4313–4324.

[62] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach," in *Proc. CVPR*, 2017, pp. 1944–1952.

[63] J. Cheng, T. Liu, K. Ramamohanarao, and D. Tao, "Learning with bounded instance and label-dependent label noise," in *Proc. ICML*, 2020, pp. 1789–1799.

[64] B. Garg and N. Manwani, "Robust deep ordinal regression under label noise," in *Proc. ACML*, 2020, pp. 782–796.

[65] W. Hu, Z. Li, and D. Yu, "Simple and effective regularization methods for training on noisily labeled data with generalization guarantee," in *Proc. ICLR*, 2020, pp. 1–18.

[66] A. K. Menon, B. Van Rooyen, and N. Natarajan, "Learning from binary labels with instance-dependent noise," *Mach. Learn.*, vol. 107, nos. 8–10, pp. 1561–1595, 2018.

[67] L. Torgo and J. Gama, "Regression using classification algorithms," *Intell. Data Anal.*, vol. 1, no. 4, pp. 275–292, Oct. 1997.

[68] A. Ghosh, H. Kumar, and P. Sastry, "Robust loss functions under label noise for deep neural networks," in *Proc. AAAI*, 2017, pp. 1919–1925.

[69] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, "Training deep neural networks on noisy labels with bootstrapping," in *Proc. ICLR*, 2015, pp. 1–11.

[70] E. Malach and S. Shalev-Shwartz, "Decoupling 'when to update' from 'how to update,'" in *Proc. NeurIPS*, 2017, pp. 960–970.

[71] L. P. Garcia, A. C. de Carvalho, and A. C. Lorena, "Noise detection in the meta-learning level," *Neurocomputing*, vol. 176, pp. 14–25, Dec. 2016.

[72] Y. Yan, Z. Xu, I. W. Tsang, G. Long, and Y. Yang, "Robust semi-supervised learning through label aggregation," in *Proc. AAAI*, 2016, pp. 2244–2250.

[73] H. Harutyunyan, K. Reing, G. Ver Steeg, and A. Galstyan, "Improving generalization by controlling label-noise information in neural network weights," in *Proc. ICML*, 2020, pp. 4071–4081.

[74] P. Chen, G. Chen, J. Ye, J. Zhao, and P.-A. Heng, "Noise against noise: Stochastic label noise helps combat inherent label noise," in *Proc. ICLR*, 2021, pp. 1–20.

[75] X. Chen and A. Gupta, "Webly supervised learning of convolutional networks," in *Proc. ICCV*, 2015, pp. 1431–1439.

[76] A. J. Bekker and J. Goldberger, "Training deep neural-networks based on unreliable labels," in *Proc. ICASSP*, 2016, pp. 2682–2686.

[77] S. Sukhbaatar, J. Bruna, M. Paluri, L. Bourdev, and R. Fergus, "Training convolutional networks with noisy labels," in *Proc. ICLRW*, 2015, pp. 1–11.

[78] I. Jindal, M. Nokleby, and X. Chen, "Learning deep networks from noisy labels with dropout regularization," in *Proc. ICDM*, 2016, pp. 967–972.

[79] J. Goldberger and E. Ben-Reuven, "Training deep neural-networks using a noise adaptation layer," in *Proc. ICLR*, 2017, pp. 1–9.

[80] B. Han *et al.*, "Masking: A new perspective of noisy supervision," in *Proc. NeurIPS*, 2018, pp. 5836–5846.

[81] J. Yao *et al.*, "Deep learning from noisy image labels with quality embedding," *IEEE Trans. Image Process.*, vol. 28, no. 4, pp. 1909–1922, Dec. 2018.

[82] L. Cheng *et al.*, "Weakly supervised learning with side information for noisy labeled images," in *Proc. ECCV*, 2020, pp. 306–321.

[83] X. Xia *et al.*, "Extended T: Learning with mixed closed-set and open-set noisy labels," 2020, *arXiv:2012.00932*.

[84] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. NeurIPS*, 2014, pp. 2672–2680.

[85] K. Lee, S. Yun, K. Lee, H. Lee, B. Li, and J. Shin, "Robust inference via generative classifiers for handling noisy labels," in *Proc. ICML*, 2019, pp. 3763–3772.

[86] R. Tanno, A. Saeedi, S. Sankaranarayanan, D. C. Alexander, and N. Silberman, "Learning from noisy labels by regularized estimation of annotator confusion," in *Proc. CVPR*, 2019, pp. 11244–11253.

[87] S. Jenni and P. Favaro, "Deep bilevel learning," in *Proc. ECCV*, 2018, pp. 618–633.

[88] D. Hendrycks, K. Lee, and M. Mazeika, "Using pre-training can improve model robustness and uncertainty," in *Proc. ICML*, 2019, pp. 2712–2721.

[89] A. K. Menon, A. S. Rawat, S. J. Reddi, and S. Kumar, "Can gradient clipping mitigate label noise?" in *Proc. ICLR*, 2020, pp. 1–26.

[90] X. Xia *et al.*, "Robust early-learning: Hindering the memorization of noisy labels," in *Proc. ICLR*, 2021, pp. 1–15.

[91] H. Wei, L. Tao, R. Xie, and B. An, "Open-set label noise can improve robustness against inherent label noise," in *Proc. NeurIPS*, 2021, pp. 1–15.

[92] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. ICLR*, 2014, pp. 1–11.

[93] S. Zhang, Y. Hou, B. Wang, and D. Song, "Regularizing neural networks via retaining confident connections," *Entropy*, vol. 19, no. 7, p. 313, Jun. 2017.

[94] M. Lukasik, S. Bhojanapalli, A. Menon, and S. Kumar, "Does label smoothing mitigate label noise?" in *Proc. ICLR*, 2020, pp. 6448–6458.

[95] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," in *Proc. ICLR*, 2018, pp. 1–13.

[96] B. C. Csáji *et al.*, "Approximation with artificial neural networks," *Fac. Sci., Etvs Lornd Univ., Hung.*, vol. 24, no. 48, p. 7, 2001.

[97] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 8778–8788.

[98] Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey, "Symmetric cross entropy for robust learning with noisy labels," in *Proc. ICCV*, 2019, pp. 322–330.

[99] Y. Lyu and I. W. Tsang, "Curriculum loss: Robust learning and generalization against label corruption," in *Proc. ICLR*, 2020, pp. 1–22.

[100] L. Feng, S. Shu, Z. Lin, F. Lv, L. Li, and B. An, "Can cross entropy loss be robust to label noise," in *Proc. IJCAI*, 2020, pp. 2206–2212.

[101] Y. Liu and H. Guo, "Peer loss functions: Learning from noisy labels without knowing noise rates," in *Proc. ICML*, 2020, pp. 6226–6236.

[102] H. Kumar, N. Manwani, and P. Sastry, "Robust learning of multi-label classifiers under label noise," in *Proc. CODS-COMAD*, 2020, pp. 90–97.

[103] E. Amid, M. K. Warmuth, and S. Srinivasan, "Two-temperature logistic regression based on the tsallis divergence," in *Proc. AISTATS*, 2019, pp. 2388–2396.

[104] E. Amid, M. K. Warmuth, R. Anil, and T. Koren, "Robust bi-tempered logistic loss based on Bregman divergences," in *Proc. NeurIPS*, 2019, pp. 14987–14996.

[105] X. Ma, H. Huang, Y. Wang, S. Romano, S. Erfani, and J. Bailey, "Normalized loss functions for deep learning with noisy labels," in *Proc. ICML*, 2020, pp. 6543–6553.

[106] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," in *Proc. ICML*, 2018, pp. 4334–4343.

[107] D. Hendrycks, M. Mazeika, D. Wilson, and K. Gimpel, "Using trusted data to train deep networks on labels corrupted by severe noise," in *Proc. NeurIPS*, 2018, pp. 10456–10465.

[108] R. Wang, T. Liu, and D. Tao, "Multiclass learning with partially corrupted labels," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2568–2580, Jun. 2018.

[109] H.-S. Chang, E. Learned-Miller, and A. McCallum, "Active bias: Training more accurate neural networks by emphasizing high variance samples," in *Proc. NeurIPS*, 2017, pp. 1002–1012.

[110] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Unsupervised label noise modeling and loss correction," in *Proc. ICML*, 2019, pp. 312–321.

[111] X. Ma *et al.*, "Dimensionality-driven learning with noisy labels," in *Proc. ICML*, 2018, pp. 3355–3364.

[112] B. Han *et al.*, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 8527–8537.

[113] X. Xia *et al.*, "Are anchor points really indispensable in label-noise learning?" in *Proc. NeurIPS*, 2019, pp. 1–12.

[114] Y. Yao *et al.*, "Dual T: Reducing estimation error for transition matrix in label-noise learning," in *Proc. NeurIPS*, 2020, pp. 7260–7271.

[115] Y. Zhang and M. Sugiyama, "Approximating instance-dependent noise via instance-confidence embedding," 2021, *arXiv:2103.13569*.

[116] S. Yang *et al.*, "Estimating instance-dependent label-noise transition matrix using DNNs," 2021, *arXiv:2105.13001*.

[117] T. Liu and D. Tao, "Classification with noisy labels by importance reweighting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 3, pp. 447–461, Mar. 2016.

[118] H. Zhang, X. Xing, and L. Liu, "DualGraph: A graph-based method for reasoning about label noise," in *Proc. CVPR*, 2021, pp. 9654–9663.

[119] L. Huang, C. Zhang, and H. Zhang, "Self-adaptive training: Beyond empirical risk minimization," in *Proc. NeurIPS*, 2020, pp. 19365–19376.

[120] M. E. Houle, "Local intrinsic dimensionality I: An extreme-value-theoretic foundation for similarity applications," in *Proc. SISAP*, 2017, pp. 64–79.

[121] S. Zheng, P. Wu, A. Goswami, M. Goswami, D. Metaxas, and C. Chen, "Error-bounded correction of noisy labels," in *Proc. ICML*, 2020, pp. 11447–11457.

[122] P. Chen, J. Ye, G. Chen, J. Zhao, and P.-A. Heng, "Beyond class-conditional assumption: A primary attempt to combat instance-dependent label noise," in *Proc. AAAI*, 2021, pp. 1–10.

[123] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. ICML*, 2017, pp. 1126–1135.

[124] J. Shu *et al.*, "Meta-weight-net: Learning an explicit mapping for sample weighting," in *Proc. NeurIPS*, 2019, pp. 1917–1928.

[125] Z. Wang, G. Hu, and Q. Hu, "Training noise-robust deep neural networks via meta-learning," in *Proc. CVPR*, 2020, pp. 4524–4533.

[126] M. Dehghani, A. Severyn, S. Rothe, and J. Kamps, "Learning to learn from weak supervision by full supervision," in *Proc. NeurIPSW*, 2017, pp. 1–8.

[127] M. Dehghani, A. Severyn, S. Rothe, and J. Kamps, "Avoiding your teacher's mistakes: Training neural networks with controlled weak supervision," 2017, *arXiv:1711.00313*.

[128] Z. Zhang, H. Zhang, S. O. Arik, H. Lee, and T. Pfister, "Distilling effective supervision from severe label noise," in *Proc. CVPR*, 2020, pp. 9294–9303.

[129] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L.-J. Li, "Learning from noisy labels with distillation," in *Proc. ICCV*, 2017, pp. 1910–1918.

[130] G. Zheng, A. H. Awadallah, and S. Dumais, "Meta label correction for noisy label learning," in *Proc. AAAI*, 2021, pp. 1–9.

[131] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, "MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in *Proc. ICML*, 2018, pp. 2304–2313.

[132] X. Yu, B. Han, J. Yao, G. Niu, I. W. Tsang, and M. Sugiyama, "How does disagreement help generalization against label corruption?" in *Proc. ICML*, 2019, pp. 7164–7173.

[133] Y. Wang *et al.*, "Iterative learning with open-set noisy labels," in *Proc. CVPR*, 2018, pp. 8688–8696.

[134] Y. Shen and S. Sanghavi, "Learning with bad training data via iterative trimmed loss minimization," in *Proc. ICML*, 2019, pp. 5739–5748.

[135] P. Chen, B. Liao, G. Chen, and S. Zhang, "Understanding and utilizing deep neural networks trained with noisy labels," in *Proc. ICML*, 2019, pp. 1062–1070.

[136] D. T. Nguyen, C. K. Mummadi, T. P. N. Ngo, T. H. P. Nguyen, L. Beggel, and T. Brox, "SELF: Learning to filter noisy labels with self-ensembling," in *Proc. ICLR*, 2020, pp. 1–15.

[137] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, "Robust learning by self-transition for handling noisy labels," in *Proc. KDD*, 2021, pp. 1490–1500.

[138] D. Krueger *et al.*, "Deep nets don't learn via memorization," in *Proc. ICLRW*, 2017, pp. 1–4.

[139] C. Zhang, S. Bengio, M. Hardt, M. C. Mozer, and Y. Singer, "Identity Crisis: Memorization and generalization under extreme overparameterization," in *Proc. ICLR*, 2020, pp. 1–39.

[140] Q. Yao, H. Yang, B. Han, G. Niu, and J. T.-Y. Kwok, "Searching to exploit memorization effect in learning with noisy labels," in *Proc. ICML*, 2020, pp. 10789–10798.

[141] H. Song, M. Kim, D. Park, and J.-G. Lee, "How does early stopping help generalization against label noise?" 2019, *arXiv:1911.08059*.

[142] J. Li, R. Socher, and S. C. Hoi, "DivideMix: Learning with noisy labels as semi-supervised learning," in *Proc. ICLR*, 2020, pp. 1–14.

[143] H. Wei, L. Feng, X. Chen, and B. An, "Combating noisy labels by agreement: A joint training method with co-regularization," in *Proc. CVPR*, 2020, pp. 13726–13735.

[144] J. Huang, L. Qu, R. Jia, and B. Zhao, "O2U-Net: A simple noisy label detection approach for deep neural networks," in *Proc. ICCV*, Oct. 2019, pp. 3326–3334.

[145] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. Int. Conf. Manage. Data (SIGMOD)*, vol. 29, 2000, pp. 93–104.

[146] P. Wu, S. Zheng, M. Goswami, D. Metaxas, and C. Chen, "A topological filter for learning with label noise," in *Proc. NeurIPS*, 2020, pp. 21382–21393.

[147] Z.-F. Wu, T. Wei, J. Jiang, C. Mao, M. Tang, and Y.-F. Li, "NGC: A unified framework for learning with open-world noisy data," in *Proc. ICCV*, 2021, pp. 62–71.

[148] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1195–1204.

[149] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "MixMatch: A holistic approach to semi-supervised learning," in *Proc. NeurIPS*, 2019, pp. 5050–5060.

[150] T. Zhou, S. Wang, and J. Bilmes, "Robust curriculum learning: From clean label detection to noisy label self-correction," in *Proc. ICLR*, 2021, pp. 1–18.

[151] X. Li, T. Liu, B. Han, G. Niu, and M. Sugiyama, "Provably end-to-end label-noise learning without anchor points," in *Proc. ICML*, 2021, pp. 6403–6413.

[152] G. Pleiss, T. Zhang, E. R. Elenberg, and K. Q. Weinberger. (2020). *Detecting Noisy Training Data With Loss Curves*. [Online]. Available: https://openreview.net/forum?id=HyenUkrtDB

[153] C. Scott, "A rate of convergence for mixture proportion estimation, with application to learning from noisy labels," in *Proc. AISTATS*, 2015, pp. 838–846.

[154] Y. LeCun, C. Cortes, and C. J. Burges. (1998). *The MNIST Database of Handwritten Digits, 1998*. [Online]. Available: http://yann.lecun.com/exdb/mnist

[155] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.

[156] A. Krizhevsky, V. Nair, and G. Hinton. (2014). *CIFAR-10 and CIFAR-100 Datasets*. [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html

[157] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NeurIPSW*, 2011, pp. 1–9.

[158] A. Karpathy *et al.*, "Cs231n convolutional neural networks for visual recognition," *Neural Netw.*, vol. 1, p. 1, Dec. 2016.

[159] J. Wei, Z. Zhu, H. Cheng, T. Liu, G. Niu, and Y. Liu, "Learning with noisy labels revisited: A study using real-world human annotations," 2011, *arXiv:2110.12088*.

[160] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101–mining discriminative components with random forests," in *Proc. ECCV*, 2014, pp. 446–461.

[161] J. Bootkrajang and J. Chaijaruwanich, "Towards instance-dependent label noise-tolerant classification: A probabilistic approach," *Pattern Anal. Appl.*, vol. 23, no. 1, pp. 95–111, 2020.

[162] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *Int. J. Data Warehousing Mining*, vol. 3, no. 3, pp. 1–13, 2007.

[163] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognit.*, vol. 37, no. 9, pp. 1757–1771, Sep. 2004.

[164] I. Krasin *et al.*, "OpenImages: A public dataset for large-scale multi-label and multi-class image classification," *Dataset*, vol. 2, no. 3, p. 18, 2017.

[165] W. Zhao and C. Gomes, "Evaluating multi-label classifiers with noisy labels," 2021, *arXiv:2102.08427*.

[166] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *J. Big Data*, vol. 6, no. 1, pp. 1–54, Dec. 2019.

[167] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," in *Proc. NeurIPS*, 2016, pp. 1–9.

[168] H. Jiang and O. Nachum, "Identifying and correcting label bias in machine learning," in *Proc. AISTATS*, 2020, pp. 702–712.

[169] S. Wang, W. Guo, H. Narasimhan, A. Cotter, M. Gupta, and M. I. Jordan, "Robust optimization for fairness with noisy protected groups," in *Proc. NeurIPS*, 2020, pp. 5190–5203.

[170] J. Wang, Y. Liu, and C. Levy, "Fair classification with group-dependent label noise," in *Proc. FAccT*, 2021, pp. 526–536.

[171] J. Uesato, J.-B. Alayrac, P.-S. Huang, R. Stanforth, A. Fawzi, and P. Kohli, "Are labels required for improving adversarial robustness?" in *Proc. NeurIPS*, 2019, pp. 12192–12202.

[172] B. B. Damodaran, K. Fatras, S. Lobry, R. Flamary, D. Tuia, and N. Courty, "Wasserstein adversarial regularization (WAR) on label noise," in *Proc. ICLR*, 2020, pp. 1–15.

[173] J. Zhu *et al.*, "Understanding the interaction of adversarial training with noisy labels," 2021, *arXiv:2102.03482*.

[174] G. Nguyen *et al.*, "Machine learning and deep learning frameworks and libraries for large-scale data mining: A survey," *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 77–124, 2019.

Hwanjun Song received the Ph.D. degree from the Graduate School of Knowledge Service Engineering, Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 2021.

He was a Research Intern with Google Research, Seoul, South Korea, in 2020. He is currently a Research Scientist with the NAVER AI Laboratory, Seongnam, South Korea. His research interest includes designing advanced approaches to handle large-scale and noisy data, including two main real-world challenges for the practical use of AI approaches.

Minseok Kim (Student Member, IEEE) received the master's degree from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 2018, where he is currently pursuing the Ph.D. degree under the supervision of Prof. Jae-Gil Lee with the Graduate School of Knowledge Service Engineering.

His current research interests include robustness and uncertainty in machine learning, data augmentation, and personalized recommendation systems.

Dongmin Park received the master's degree from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 2020, where he is currently pursuing the Ph.D. degree under the supervision of Prof. Jae-Gil Lee with the Graduate School of Knowledge Service Engineering.

His current research interests include robust deep learning and representation learning in graph and time–series data.

Yooju Shin received the master's degree from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 2019, where he is currently pursuing the Ph.D. degree under the supervision of Prof. Jae-Gil Lee with the Graduate School of Knowledge Service Engineering.

His current research interests include out-of-distribution learning and label efficient learning in time series.

Jae-Gil Lee (Member, IEEE) received the Ph.D. degree in computer science from the Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 2005.

He was a Post-Doctoral Researcher with the IBM Research—Almaden Laboratory, San Jose, CA, USA, and a Post-Doctoral Research Associate with the University of Illinois at Urbana–Champaign, Champaign, IL, USA. He is currently an Associate Professor with KAIST, where he is also the Leader of the Data Mining Laboratory. His research interests include mobility and stream data mining, deep learning-based big data analysis, and distributed deep learning.