Learning from Noisy Labels with Deep Neural Networks: A Survey

Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, Jae-Gil Lee

***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 57 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. All the contents will be available at https://github.com/songhwanjun/Awesome-Noisy-Labels.**

***Index Terms*—deep learning, noisy label, label noise, robust optimization, robust deep learning, classification, 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 non-expert 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 source often results in unreliable labels [9]–[11]. In addition, data labels can be extremely complex even for **R1:** experienced domain experts [12], [13]; they can also be adversarially manipulated by a label-flipping attack [14]. 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% [15]–[18].

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 [19], [20]. Zhang et al. [21] 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 [22], weight decay [23], dropout [24], and batch normalization [25] have been applied popularly, but they do *not* completely overcome the overfitting issue by themselves. As shown in Figure 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. Additionally, the accuracy drop with label noise is considered to be more harmful than with other noises, such as input noise [26]. Hence, achieving a good generalization capability in the presence of noisy labels is a key challenge.

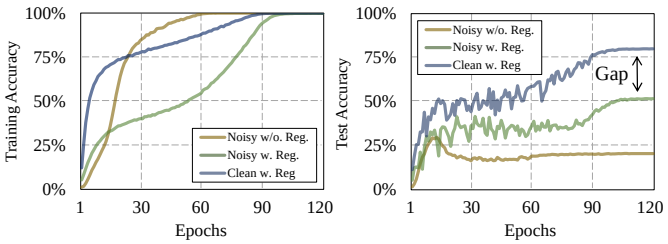


Fig. 1. Convergence curves of training and test accuracy when training WideResNet-16-8 using a standard training method on the CIFAR-100 dataset with the symmetric noise of 40%: "Noisy w/o. Reg." and "Noisy w. Reg." are the models trained on noisy data without and with regularization, respectively, and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

Several studies have been conducted to investigate supervised learning under noisy labels. Beyond conventional machine learning techniques [12], [27], 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 162 recent studies by recursively tracking relevant bibliographies in papers published at premier research conferences, such as CVPR, ICCV, NeurIPS, ICML, 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 Figure 2 (see Section III for details).

A. Related Surveys

Frénay and Verleysen [12] discussed the potential negative consequence of learning from noisy labels and provided a comprehensive survey on noise-robust classification methods,

H. Song is with NAVER AI Lab, Seongnam 13561, Republic of Korea (e-mail: ghkswns91@gmail.com); M. Kim, D, Park, Y, Shin, and J.-G., Lee are with the Graduate School of Knowledge Service Engineering, Korea Advanced Institute of Science and Technology, Daejeon 34141, Republic of Korea (e-mail: minseokkim@kaist.ac.kr; dongminpark@kaist.ac.kr; yooju24@kaist.ac.kr; jaegil@kaist.ac.kr).

Manuscript received March 31, 2021.

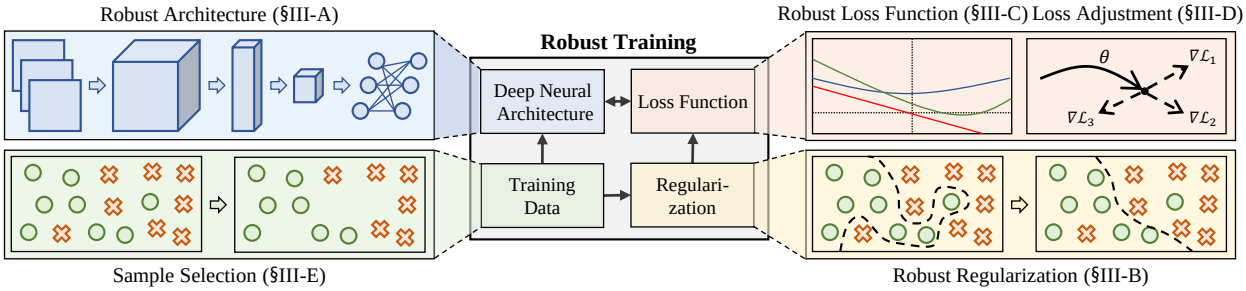


Fig. 2. Categorization of recent deep learning methods for overcomming noisy labels.

focusing on conventional supervised approaches such as naïve Bayes and support vector machines. Furthermore, their survey included the definitions and sources of label noise as well as the taxonomy of label noise. Zhang et al. [27] discussed another aspect of label noise in crowdsourced data annotated by non-experts and provided a thorough review of expectation-maximization (EM) algorithms that were proposed to improve the quality of crowdsourced labels. Meanwhile, Nigam et al. [28] 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 Figure 2. Recently, Han et al. [29] 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, but they rather focus on more general views, such as input data, objective function, and optimization policies. Furthermore, this survey is the first to present a comprehensive methodological comparison of existing robust training approaches (see Table II and Table III).

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 [30], [31]. *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 [32]–[35]. Meanwhile, *data imputation* primarily deals with missing inputs in training data, where missing values are estimated from the observed ones [31], [36]. 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*.

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

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 label transition matrix

[37]. In this paper, 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 $\mathcal{R}_{\mathcal{D}}(f)$,

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

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 $\mathcal{B}_t$,

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

where η is a learning rate specified.

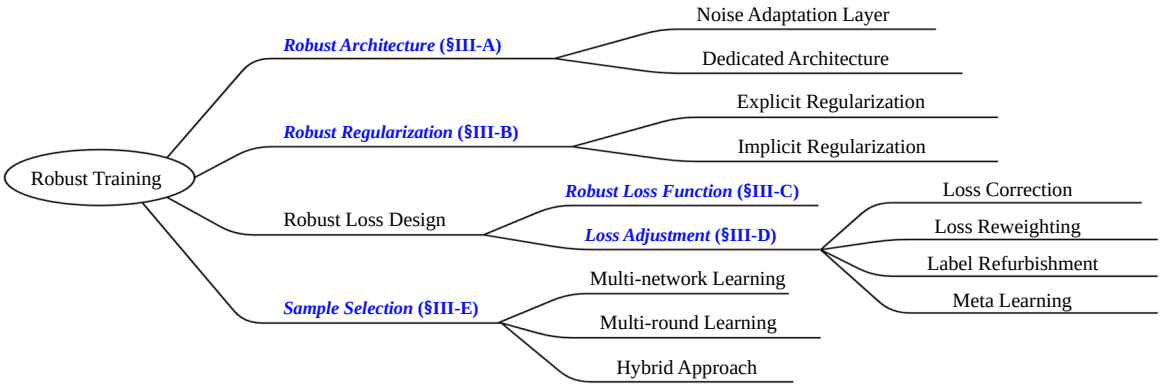


Fig. 3. A high level research overview of robust deep learning for noisy labels. The research directions that are actively contributed by the machine learning community are categorized into five groups in blue italic.

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 [12], [27], [28]. Hence, mitigating the adverse effects of noisy labels is essential to enable noise-tolerant training for deep learning.

B. Taxonomy of Label Noise

R2: 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 [12]. **R2:** 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 [21], [38]. That is, the true label is corrupted by a *label 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 $\forall_{i=j}\mathrm{T}_{ij} = 1-\tau \wedge \forall_{i\neq j}\mathrm{T}_{ij} = \frac{\tau}{c-1}$, 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 [15], [39]. 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. Non-deep 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 [12], [28], [40], as follows:

- **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 [41], [42]. 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 [43]–[45]. Nevertheless, this family of methods suffers from over-cleaning issue that overly removes even the true-labeled examples.
- **Surrogate Loss:** Motivated by the noise-tolerance of the 0-1 loss function [38], many researchers have attempted to resolve its inherent limitations, such as computational hardness and non-convexity 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 [46]–[50]. However, these loss functions cannot support the multi-class classification task.
- **Probabilistic Method:** Under the assumption that the distribution of features is helpful in solving the problem of learning from noisy labels [51], the confidence of each label is estimated by clustering and then used for a weighted training scheme [52]. This confidence is also used to convert hard labels into soft labels to reflect the uncertainty of labels [53]. 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 [54]. However, this family of methods may exacerbate the overfitting issue owing to the increased number of model parameters.
- **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 the robustness of them. To develop a robust SVM model, misclassified examples during learning are penalized in the objective [55],

[56]. 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 [57], [58]. However, it is infeasible to apply their design principles to deep learning.

D. Theoretical Foundations

Deep learning is more susceptible to label noises than traditional machine learning owing to its high expressive power, as proven by many researchers [20], [59], [60]. There has been significant effort to understand why noisy labels negatively affect the performance of DNNs [21], [60]–[62]. This theoretical understanding has led to the architectural or algorithmic design which is more robust than the non-deep learning methods. A detailed analysis of theoretical understanding for robust deep learning was provided by Han et al. [29]; *three* high-level perspectives have been widely leveraged to design robust approaches, as follows:

- **Label Transition**: From the view of training data, the noise process is modeled by the *label transition matrix* T, which can discover the underlying noise transition pattern. Given an example x, the estimation of the noisy label distribution for the example x is expressed by

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

  Thus, the notion of the label transition matrix has been employed to build robust approaches via a noise adaptation layer [24], [63], [64] or loss correction [61], [65], both of which aim to calibrate the output of corrupted DNNs.

- **Risk Minimization**: 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 [49], with probability at least $1-\delta$, there exists a non-decreasing function ζ_ℓ with $\zeta_\ell(0)=0$ [38] such that

$$\mathcal{R}_{\mathcal{D}}(\hat{f}) - \mathcal{R}^* \leq \overbrace{\zeta_\ell\Big(\min_{f\in\mathcal{F}}\mathcal{R}_{\ell,\mathcal{D}}(f) - \min_f \mathcal{R}_{\ell,\mathcal{D}}(f) + 4L_p\mathrm{RC}(\mathcal{F}) + 2\sqrt{\log(1/\delta)/2|\mathcal{D}|}\Big)}^{\text{Approximation and Estimation Errors}}, \tag{4}$$

  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 [66], the Bayes optimal classifier f^* is guaranteed to be in the hypothesis class $\mathcal{F}$ with DNNs. Thus, robust loss functions for DNNs have been designed using this theoretical foundation [67]–[70].

- **Memorization Effect**: The *memorization nature* of DNNs was explored theoretically in recent literature [71]–[73]. Assuming clusterable data where the clusters are located on the unit Euclidean ball, Li et al. [60] proved the distance from the initial weight W_0 to the weight W_t after t iterations,

$$\|W_t - W_0\|_F \lesssim \big(\sqrt{K} + (K^2\epsilon_0/\|C\|^2)t\big), \tag{5}$$

  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. Eq. (5) 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 the beginning of training [29], [60]. In the empirical studies [20], [74], this result is also known as the *memorization effect* that DNNs tend to first learn simple and generalized patterns and then gradually overfit to all the noisy patterns. Thus, to achieve better generalization, early stopping [60], [74], [75] and favoring small-loss training examples [76]–[80] are commonly employed to design robust training methods.

E. Regression with Noisy Labels

R3: Regression is another sub-field 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 [81], and thus it learns the mapping function $f(\ \cdot\ ;\Theta) : \mathcal{X} \to \mathcal{Y}$, 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 noises are considered in the regression task. An *additive noise* [82] is formulated by $\tilde{y} := y + \epsilon$ where ϵ is drawn from a random distribution independent from the input feature; an *instance-dependent noise* [83] is formulated by $\tilde{y} := \rho(x)$ where $\rho : \mathcal{X} \to \mathcal{Y}$ 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 [84]. Thus, in this survey, we focus on the classification setting for which 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 [15], [24], [67], [85]–[90]. Figure 3 shows an overview of recent research directions conducted by the machine learning community. **R1:** All of them (i.e., §III-A – §III-E) focused on making a supervised learning process more robust to label noise:

- (§III-A) Robust architecture: 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 noises;
- (§III-B) Robust regularization: enforcing a DNN to overfit less to false-labeled examples explicitly or implicitly;
- (§III-C) Robust loss function: improving the loss function;
- (§III-D) Loss adjustment: adjusting the loss value according to the confidence of a given loss (or label) by loss correction, loss reweighting, or label refurbishment;
- (§III-E) Sample selection: identifying true-labeled examples from noisy training data via multi-network or multi-round learning.

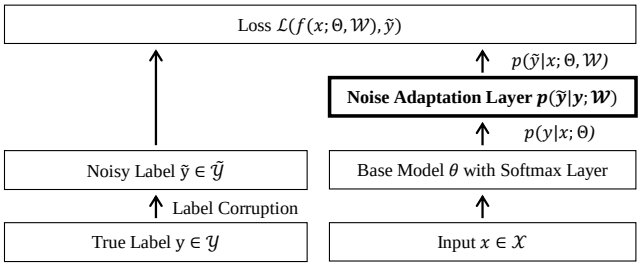


Fig. 4. Noise modeling process using the noise adaptation layer.

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

Figure 2 illustrates the categorization of robust training methods using these five groups.

A. Robust Architecture

In numerous studies, architectural changes have been made to model the label transition matrix of a noisy dataset [15], [63], [64], [91]–[96]. 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**:* The noise adaptation layer is intended to mimic the noise behavior (i.e., label transition) in learning a DNN. Let $p(y|x;\Theta)$ be the output of the base DNN with a softmax output layer. Then, similar to Eq. (3), the probability of an example x being predicted as its annotated noisy label $\tilde{y}$ is parameterized by

$$\begin{aligned} p(\tilde{y}=j|x;\Theta,\mathcal{W}) &= \sum_{i=1}^{c} p(\tilde{y}=j, y=i|x;\Theta,\mathcal{W}) \\ &= \sum_{i=1}^{c} \underbrace{p(\tilde{y}=j|y=i;\mathcal{W})}_{\text{Noise Adaptation Layer}} \underbrace{p(y=i|x;\Theta)}_{\text{Base Model}}. \end{aligned} \tag{6}$$

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

Technical Detail: *Webly learning* [91] 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* [63] initializes $\mathcal{W}$ to an identity matrix and adds a regularizer to force $\mathcal{W}$ to diffuse during DNN training. The *dropout noise model* [24] applies dropout regularization to the adaptation layer, whose output is normalized by the softmax function to implicitly diffuse $\mathcal{W}$. The *s-model* [64] is similar to the *dropout noise model* but dropout is not applied. The *c-model* [64] 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* [92] adopts the EM algorithm to iterate the E-step to estimate the label transition matrix and the M-step to back-propagate the DNN.

$\mathbf{R1}$: 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 [97]. Meanwhile, for the EM-based method, becoming stuck in local optima is inevitable, and high computational costs are incurred [64].

*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 [15], [94], [95]. 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* [15] 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 pre-training step with a small amount of clean data. Meanwhile, *masking* [94] 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) [98] was employed in this study. Recently, the *contrastive-additive noise network* [95] 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* [99] builds a simple yet robust generative classifier on top of any discriminative DNN pre-trained 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 [22]–[25]. By avoiding overfitting in model training, the robustness to label noise improves with widely-used regularization techniques such as *data augmentation* [22], *weight decay* [23], *dropout* [24], and *batch normalization* [25]. $\mathbf{R1}$: 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 [100]. 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* [101] 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* [100] 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, *pre-training* [102] empirically proves that fine-tuning on a pre-trained model provides a significant improvement in robustness compared with models trained from scratch; the universal representations of pre-training prevent the model parameters from being updated in the wrong direction by noisy labels. *PHuber* [103] proposes a composite loss-based gradient clipping, which is a variation of standard gradient clipping for label noise robustness. *Robust early-learning* [104] classifies critical parameters and non-critical parameters for fitting clean and noise labels, respectively. Then, it penalizes only the non-critical ones with a different update rule.

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* [105] enhances the noise tolerance by encouraging the DNN to correctly classify both original inputs and hostilely perturbed ones. *Label smoothing* [106], [107] 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,

$$\begin{gathered}\bar{y} = \big\langle \bar{y}(1), \bar{y}(2), \ldots, \bar{y}(c) \big\rangle, \\ \text{where } \bar{y}(i) = (1-\alpha)\cdot[\tilde{y}=i] + \alpha/c \text{ and } \alpha \in [0,1].\end{gathered} \tag{7}$$

Here, $[\cdot]$ is the iverson bracket and α is the smoothing degree. In contrast, *mixup* [108] 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 $\tilde{\mathcal{D}}$,

$$x_{mix} = \lambda x_i + (1-\lambda)x_j \quad \text{and} \quad y_{mix} = \lambda \tilde{y}_i + (1-\lambda)\tilde{y}_j, \tag{8}$$

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, extended feature or label space slows down the convergence of training.

C. Robust Loss Function

Considering the robustness of risk minimization schemes on the loss function, researchers have attempted to design robust loss functions [67]–[70], [109], [110]. The goal is to provide a loss function that achieves a small risk for unseen clean data even when noisy labels exist in the training data.

Technical Detail: Initially, Manwani and Sastry [47] 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 multi-class classification using deep learning [67]. 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 $\tau < \frac{c-1}{c}$ and

$$\sum_{j=1}^{c} \ell\big(f(x;\Theta), y=j\big) = C, \ \forall x \in \mathcal{X}, \ \forall f, \tag{9}$$

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 *multi-label* classification was provided by Kumar et al. [111]. 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* [67] 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) [68] 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. Inspired by the symmetricity of the Kullback-Leibler divergence, the symmetric cross entropy (SCE) [69] was proposed by combining a noise tolerance term, namely reverse cross entropy loss, with the standard CCE loss.

Meanwhile, the *curriculum loss* (CL) [70] is a surrogate loss of the 0-1 loss function; it provides a tight upper bound and can easily be extended to multi-class classification. The *active passive loss* (APL) [112] 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 [113]. Moreover, the modification of the loss function increases the training time for convergence [68].

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 [18], [61], [65], [85], [114]–[117]. The methods associated with it can be categorized into three groups depending on their adjustment philosophy: *1) loss correction* that estimates the label 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. $\mathbf{R3}$: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 [76].

*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* [61] initially approximates the label 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 a 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 $y \in \{1, \dots, c\}$, 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, $\mathbf{R1}$:

$$\overleftarrow{\ell}\big(f(x;\Theta),\tilde{y}\big) = \hat{\mathrm{T}}^{-1}\Big\langle \ell\big(f(x;\Theta),1\big),\dots,\ell\big(f(x;\Theta),c\big)\Big\rangle^{\top}, \quad (10)$$

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

Conversely, *forward correction* [61] uses a linear combination of a DNN's softmax outputs before applying the loss function. Hence, the forward correction $\vec{\ell}$ is performed by multiplying the estimated transition probability with the softmax outputs during the forward propagation step, $\mathbf{R1}$:

$$\begin{aligned}\vec{\ell}\big(f(x;\Theta),\tilde{y}\big) &= \ell\Big(\Big\langle \hat{p}(\tilde{y}|1),\dots,\hat{p}(\tilde{y}|c)\Big\rangle f(x;\Theta)^{\top},\tilde{y}\Big)\\ &= \ell\big(\hat{\mathrm{T}}^{\top} f(x;\Theta)^{\top},\tilde{y}\big).\end{aligned} \quad (11)$$

Furthermore, *gold loss correction* [65] 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* [118] provides a solution that can infer the transition matrix without anchor points, and *Dual T* [119] factorizes the matrix into the product of two easy-to-estimate matrices to avoid directly estimating the noisy class posterior.

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 [120], 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, $\mathbf{R1}$:

$$\Theta_{t+1} = \Theta_t - \eta\nabla\Big(\frac{1}{|\mathcal{B}_t|}\sum_{(x,\tilde{y})\in\mathcal{B}_t}\overbrace{w(x,\tilde{y})\ell\big(f(x;\Theta_t),\tilde{y}\big)}^{\text{Reweighted Loss}}\Big), \quad (12)$$

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* [114], the ratio of two joint data distributions $w(x, y) = P_{\mathcal{D}}(x, \tilde{y})/P_{\tilde{\mathcal{D}}}(x, \tilde{y})$ 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* [115] emphasizes uncertain examples with inconsistent label predictions by assigning their prediction variances as the weights for training.

Remark: These approaches need to manually pre-specify the weighting function as well as its additional hyper-parameters, 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 $\hat{y}$,

$$y^{refurb} = \alpha\tilde{y} + (1-\alpha)\hat{y}, \quad (13)$$

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* [85] 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* [116] dynamically adjusts the confidence α of individual training examples. The

confidence α is obtained by fitting a two-component and one-dimensional beta mixture model to the loss distribution of all training examples. *Self-adaptive training* [121] applies the exponential moving average to alleviate the instability issue of using instantaneous prediction of the current DNN,

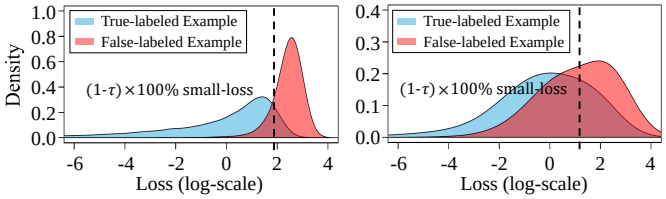
$$y_{t+1}^{refurb} = \alpha y_t^{refurb} + (1-\alpha)\hat{y}, \text{ where } y_0^{refurb} = \tilde{y} \quad (14)$$

D2L [117] trains a DNN using a dimensionality-driven learning strategy to avoid overfitting to false labels. A simple measure called *local intrinsic dimensionality* [122] 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* [18] 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* [123] selectively refurbishes the label of noisy examples, but a theoretical error-bound is provided. Alternatively, *SEAL* [124] averages the softmax output of a DNN on each example over the whole training process, then re-trains 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 to wrongly refurbished labels.

*4) **Meta Learning**:* [R1] In recent years, meta learning becomes an important topic in the machine learning community and is applied to improve noise robustness [125]–[127]. 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 Eq. (12), the goal is to learn the weight function $w(x, \tilde{y})$. Specifically, *L2LWS* [128] and *CWS* [129] 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 are shared and jointly trained to benefit from each other. *Automatic reweighting* [113] 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* [126] parameterizes the weighting function as a multi-layer perceptron network with only one hidden layer. A meta-objective is defined to update its parameters such



(a) Symmetric Noise 40%. (b) Asymmetric Noise 40%.

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. [74].)

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) [130] 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 Eq. (13), *knowledge distillation* [131] 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* [132] 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 bi-level 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 [18], [70], [76]–[79], [86], [133]–[135] have adopted sample selection that involves selecting true-labeled examples from a noisy training dataset. In this case, the update equation in Eq. (2) is modified to render a DNN more robust for noisy labels. Let $\mathcal{C}_t \subseteq \mathcal{B}_t$ be the selected *clean* examples at time t. Therefore, the DNN is updated only for the selected clean examples $\mathcal{C}_t$,

$$\Theta_{t+1} = \Theta_t - \eta \nabla \Big(\frac{1}{|\mathcal{C}_t|} \sum_{(x,\tilde{y}) \in \mathcal{C}_t} \ell\big(f(x;\Theta_t), \tilde{y}\big) \Big). \quad (15)$$

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

This training scheme is well motivated and works well in general, but it still suffers from accumulated error caused by incorrect selection. Hence, recent approaches often leverage multiple DNNs to cooperate with one another [76] or run multiple training rounds [134]. Moreover, to benefit from even false-labeled examples, loss correction or semi-supervised learning have been recently combined with the sample selection strategy [18], [80].

*1) **Multi-network Learning**:* Collaborative learning and co-training are widely used for the multi-network 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.

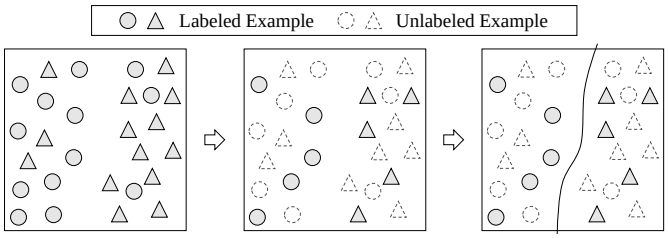
Technical Detail: Initially, *Decouple* [86] proposes the decoupling of when to update from how to update. Hence, two DNNs are maintained simultaneously and updated only the examples selected based on a disagreement between the two DNNs. Next, due to the memorization effect of DNNs, many researchers 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 illustrated in Figure 5(a). In *MentorNet* [77], a pre-trained 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* [76] and *Co-teaching+* [133] 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*.

Remark: The co-training methods help reduce the confirmation bias [76], 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 Figure 5(b).

*2) **Multi-round Learning**:* Without maintaining additional DNNs, multi-round learning iteratively refines the selected set of clean examples by repeating the training round. Thus, the selected set keeps improved as the number of rounds increases.

Technical Detail: *ITLM* [79] 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 are used to retrain the DNN in the next round. *INCV* [78] 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* [136] repeats the whole training process with the cyclical learning rate until enough loss statistics of every examples are gathered. Next, the DNN is re-trained 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* [134] detects false-labeled examples by employing the local outlier factor algorithm [137]. With a Siamese network, it gradually pulls away false-labeled examples from true-labeled samples in the deep feature space. *MORPH* [75] 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* [138] utilizes the spatial topological pattern of learned representations to detect true-labeled examples, not relying on the prediction of the noisy classifier.



(a) Noisy Data. (b) Transformed Data. (c) SSL.

Fig. 6. $\mathbf{R3}$: Procedures for semi-supervised learning under label noise.

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

*3) **Hybrid Approach**:* $\mathbf{R1}$: An inherent limitation of sample selection is to discard all the *unselected* training examples, thus resulting in a *partial* exploration of training data. To exploit all the noisy examples, researchers have attempted to combine sample selection with other orthogonal ideas.

Technical Detail: The most prominent methods in this direction is combining a specific sample selection strategy with a specific semi-supervised learning model. As illustrated in Figure 6, selected examples are treated as labeled clean data, whereas the remaining examples are treated as unlabeled. Subsequently, semi-supervised learning is performed using the transformed data. *SELF* [135] is combined with a semi-supervised learning approach to progressively filter out false-labeled examples from noisy data. By maintaining the running average model called the *mean-teacher* [139] 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* [80] leverages two-component and one-dimensional Gaussian mixture model to distinguish clean and noisy data, and then use a semi-supervised technique called *MixMatch* [140]. Recently, *RoCL* [141] employs two-phase learning strategies: supervised training on selected clean examples and then semi-supervised 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* [18] 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 *curriculum loss (CL)* [70] is combined with the robust loss function approach and used to extract the true-labeled examples from noisy data based on a manually specified selection threshold.

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.

TABLE II
COMPARISON OF ALL PROPOSED DEEP LEARNING METHODS FOR OVERCOMING NOISY LABELS.

Category		Method	P1	P2	P3	P4	P5	P6	Implementation
Robust Architecture (§III-A)	Noisy Adaptation Layer	*Webly Learning* [91]	△	×	○	○	×	×	Official (Caffe)[1]
		Noise Model [63]	△	○	○	○	×	×	Unofficial (Keras)[2]
		Dropout Noise Model [93]	△	○	○	○	×	×	Official (MATLAB)[3]
		S-model [64]	△	○	○	○	×	×	Official (Keras)[4]
		C-model [64]	△	○	○	○	×	○	Official (Keras)[4]
		NLNN [92]	△	○	○	○	×	×	Unofficial (Chainer)[5]
	Dedicated Architecture	*Probablistic Noise Model* [15]	×	×	○	×	△	○	Official (Caffe)[6]
		Masking [94]	×	○	○	×	△	○	Official (TensorFlow)[7]
		Contrastive-Additive Noise Network [95]	×	○	○	○	△	○	N/A
		RoG [99]	○	×	○	○	○	△	Official (PyTorch)[8]
Robust Regularization (§III-B)	Explicit Regularization	*Bilevel Learning* [101]	○	○	○	×	△	△	Official (TensorFlow)[9]
		Annotator Confusion [100]	○	×	○	○	△	△	Official (TensorFlow)[10]
		Pre-training [102]	○	×	○	○	△	△	Official (PyTorch)[11]
		PHuber [103]	○	○	○	○	△	△	Unofficial (PyTorch)[12]
		Robust Early-learning [104]	○	○	○	○	△	△	Official (PyTorch)[13]
	Implicit Regularization	*Adversarial Training* [105]	○	○	○	○	×	△	Unofficial (PyTorch)[14]
		Label Smoothing [106]	○	○	○	○	×	△	Unofficial (PyTorch)[15]
		Mixup [108]	○	○	○	○	×	△	Official (PyTorch)[16]
Robust Loss Function (§III-C)		*Robust MAE* [67]	○	○	○	○	×	×	N/A
		Generalized Cross Entropy [68]	○	○	○	○	×	×	Unofficial (PyTorch)[17]
		Symmetric Cross Entropy [69]	○	○	○	○	×	×	Official (Keras)[18]
		Curriculum Learning [70]	○	○	○	×	○	△	N/A
		Active Passive Loss [112]	○	○	○	○	○	△	Official (PyTorch)[19]
Loss Adjustment (§III-D)	Loss Correction	*Backward Correction* [61]	○	○	○	×	×	×	Official (Keras)[20]
		Forward Correction [61]	○	○	○	×	×	×	Official (Keras)[20]
		Gold Loss Correction [65]	○	×	○	×	×	×	Official (PyTorch)[21]
		T-revision [118]	○	×	○	○	×	×	Official (PyTorch)[22]
		Dual T [119]	○	×	○	○	△	×	N/A
	Loss Reweigting	*Importance Reweighting* [114]	○	○	○	○	×	△	Unofficial (PyTorch)[23]
		Active Bias [115]	○	○	○	○	×	△	Unofficial (TensorFlow)[24]
	Label Refurbishment	*Bootstrapping* [85]	○	○	○	×	×	△	Unofficial (Keras)[25]
		Dynamic Bootstrapping [116]	○	○	○	○	×	△	Official (PyTorch)[26]
		Self-adaptive Training [121]	○	×	○	○	○	△	Official (PyTorch)[27]
		D2L [117]	○	○	○	○	×	△	Official (Keras)[28]
		AdaCorr [123]	○	○	○	○	○	△	Official (PyTorch)[29]
		SEAL [124]	○	×	○	○	○	○	Official (PyTorch)[30]
	Meta Learning	*L2LWS* [128]	×	○	○	×	△	△	Unofficial (TensorFlow)[31]
		CWS [129]	×	○	○	×	△	△	N/A
		Automatic Reweighting [113]	○	○	○	×	△	△	Official (TensorFlow)[32]
		Meta-weight-net [126]	△	○	○	×	△	△	Official (PyTorch)[33]
		Data Coefficients [130]	○	○	○	×	△	△	Official (TensorFlow)[34]
		Knowledge Distillation [131]	○	×	○	×	△	△	N/A
		MLC [132]	○	○	○	×	△	○	Official (PyTorch)[35]
Sample Selection (§III-E)	Multi-Network Learning	*Decouple* [86]	○	○	×	○	×	△	Official (TensorFlow)[36]
		MentorNet [77]	×	×	×	×	○	△	Official (TensorFlow)[37]
		Co-teaching [76]	○	○	×	×	○	△	Official (PyTorch)[38]
		Co-teaching+ [133]	○	○	×	×	○	△	Official (PyTorch)[39]
	Multi-Round Learning	*ITLM* [79]	○	○	×	×	○	△	Official (GluonCV)[40]
		INCV [78]	○	○	×	○	○	△	Official (Keras)[41]
		O2U-Net [136]	○	○	×	×	○	△	Unofficial (PyTorch)[42]
		Iterative Detection [134]	○	○	×	○	○	△	Official (Keras)[43]
		MORPH [75]	○	○	×	○	○	△	N/A
		TopoFilter [138]	○	○	×	○	○	△	Official (PyTorch)[44]
	Hybrid Approach	*SELFIE* [18]	○	○	○	×	○	△	Official (TensorFlow)[45]
		SELF [135]	○	○	○	○	○	△	N/A
		DivideMix [80]	○	○	○	○	○	△	Official (PyTorch)[46]
		RoCL [141]	○	○	○	○	○	△	N/A

[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

IV. Methodological Comparison

In this section, we compare the 57 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 [18], [76]. 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.

- **(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.
- **(P2) No Pre-traing:** A typical approach to improve noise robustness is to use a pre-trained network; however, this incurs an additional computational cost to the learning process. "No Pre-training" ensures that the proposed method can be trained from scratch without any pre-training.
- **(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 semi-supervised learning.
- **(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.
- **(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.
- **(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, **R2:** 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, P3, and 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 "△".

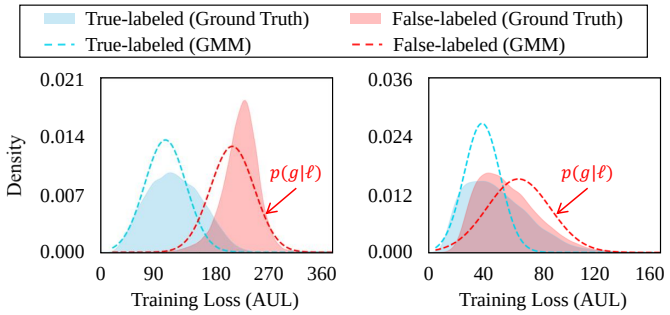
V. Noise Rate Estimation

R2: The estimation of noise rate is an imperative part of utilizing robust methods for better practical use, especially with

[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://https://github.com/sarathknv/adversarial-examples-pytorch
[15] https://github.com/CoinCheung/pytorch-loss
[16] https://github.com/facebookresearch/mixup-cifar10
[17] https://github.com/AlanChou/Truncated-Loss
[18] https://github.com/YisenWang/symmetric_cross_entropy
[19] https://github.com/HanxunH/Active-Passive-Losses
[20] https://github.com/giorgiop/loss-correction
[21] https://github.com/mmazeika/glc
[22] https://github.com/xiaoboxia/T-Revision
[23] https://github.com/xiaoboxia/Classification-with-noisy-labels
[24] https://github.com/songhwanjun/ActiveBias
[25] https://github.com/dr-darryl-wright/Noisy-Labels-with-Bootstrapping
[26] https://github.com/PaulAlbert31/LabelNoiseCorrection
[27] https://github.com/LayneH/self-adaptive-training
[28] https://github.com/xingjunm/dimensionality-driven-learning
[29] https://github.com/pingqingsheng/LRT
[30] https://github.com/chenpf1025/IDN
[31] https://github.com/krayush07/learn-by-weak-supervision
[32] https://github.com/uber-research/learning-to-reweight-examples
[33] https://github.com/xjtushujun/meta-weight-net
[34] https://github.com/google-research/google-research/tree/master/ieg
[35] https://aka.ms/MLC
[36] https://github.com/emalach/UpdateByDisagreement
[37] https://github.com/google/mentornet
[38] https://github.com/bhanML/Co-teaching
[39] https://github.com/bhanML/coteaching_plus
[40] https://github.com/yanyao-shen/ITLM-simplecode
[41] https://github.com/chenpf1025/noisy_label_understanding_utilizing
[42] https://github.com/hjimce/O2U-Net
[43] https://github.com/YisenWang/Iterative_learning
[44] https://github.com/pxiangwu/TopoFilter
[45] https://github.com/kaist-dmlab/SELFIE
[46] https://github.com/LiJunnan1992/DivideMix

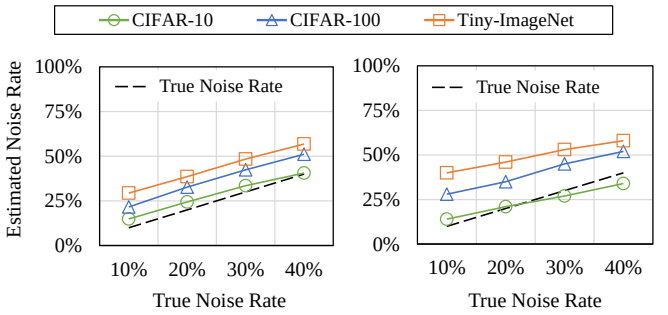
TABLE III
COMPARISON OF ROBUST DEEP LEARNING CATEGORIES FOR OVERCOMING NOISY LABELS.

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	○	○	○	○	○	△



(a) Symmetric Noise. (b) Asymmetric Noise.

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) GMM. (b) Cross Validation.

Fig. 8. Noise rate estimation using the Gaussian mixture model and cross validation when training a WideResNet-16-8 on three datasets with symmetric noise.

the methods belonging to the sample selection. The estimated noise rate is widely used to determine how many examples should be selected as clean ones [18], [76], [78]. However, comparative analysis has yet to be performed properly, though their robustness relies on the accuracy of noise rate estimation. In this section, we explore two representative approaches and compare them in terms of estimation performance.

A. Gaussian Mixture Model (GMM)

The first method is exploiting a one-dimensional and two-component GMM to model the loss distribution of true-labeled and false-labeled examples [116], [142]. As shown in Figure 7, since the loss distribution tends to be *bi-modal*, the two Gaussian components are fitted 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,

$$\hat{\tau} = \mathbb{E}_{(x,\tilde{y})\in\tilde{\mathcal{D}}}\Big[\, p\big(g \,|\, \ell\big(f(x;\Theta_t),\tilde{y}\big)\big)\,\Big], \tag{16}$$

where g is the Gaussian component with a larger loss. However, Pleiss et al. [142] 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

$$\hat{\tau} = \mathbb{E}_{(x,\tilde{y})\in\tilde{\mathcal{D}}}\Big[\, p\big(g \,|\, \mathrm{AUL}_t(x,\tilde{y})\big)\,\Big],$$
$$\text{where } \mathrm{AUL}_t(x,\tilde{y}) = \sum_{i=1}^{t} \ell\big(f(x;\Theta_t),\tilde{y}\big). \tag{17}$$

B. Cross Validation

The second method is estimating the noise rate by applying cross validation, which typically requires clean validation data [18], [76], [133]. However, such clean validation data is hard to acquire in real-world applications. Thus, Chen et al. [78] leveraged two randomly divided noisy training datasets for cross validation. Under the assumption that the two datasets share exactly the same label transition matrix, the noise rate quantifies the test accuracy of DNNs that are respectively trained and tested on the two divided sets,

$$\text{Test Accuracy} = \begin{cases} (1-\hat{\tau})^2 + \hat{\tau}^2/(c-1) & \text{if symmetric} \\ (1-\hat{\tau})^2 + \hat{\tau}^2 & \text{if asymmetric.} \end{cases} \tag{18}$$

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

C. Comparison of Noise Rate Estimation

To compare the estimation performance of using the GMM and cross validation, we trained a WideResNet-16-8 for three

TABLE IV
SUMMARY OF PUBLICLY AVAILABLE DATASETS USED FOR STUDYING LABEL NOISE.

	Dataset	# Training	# Validation	# Testing	# Classes	Noise Rate (%)
Clean Data	MNIST [143][47]	60K	N/A	10K	10	≈ 0.0
	Fashion-MNIST [144][48]	60K	N/A	10K	10	≈ 0.0
	CIFAR-10 [145][49]	50K	N/A	10K	10	≈ 0.0
	CIFAR-100 [145][49]	50K	N/A	10K	100	≈ 0.0
	SVHN [146][50]	73K	N/A	26K	10	≈ 0.0
	Tiny-ImageNet [147][52]	100K	10K	10K	200	≈ 0.0
	ImageNet [1][51]	1.3M	50K	50K	1000	≈ 0.0
Real-world Noisy Data	ANIMAL-10N [18][53]	50K	N/A	5K	10	≈ 8.0
	Food-101N [17][54]	310K	5K	25K	101	≈ 18.4
	Clothing1M [15][55]	1M	14K	10K	14	≈ 38.5
	WebVision [16][56]	2.4M	50K	50K	1000	≈ 20.0

benchmark datasets with varying noise rates. The results are plotted in Figure 8. Generally, both methods performed well on the easy dataset (i.e., CIFAR-10), but their performance worsened as the training difficulty increased from CIFAR-10 to Tiny-ImageNet because the true-labeled but hard examples are not clearly distinguishable from the false-labeled ones. Nevertheless, the GMM method showed considerably better performance than the cross validation method even in the two difficult datasets, CIFAR-100 and Tiny-ImageNet. Overall, this empirical analysis will be helpful for practitioners or researchers who design robust algorithms for noisy labels.

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 [18], [80], [134], *seven* clean datasets are widely used: MNIST[47], classification of handwritten digits [143]; Fashion-MNIST[48], classification of various clothing [144]; CIFAR-10[49] and CIFAR-100[49], classification of a subset of 80 million categorical images [145]; SVHN[50], classification of house numbers in Google Street view images [146]; ImageNet[51] and Tiny-ImageNet[52], image database organized according to the WordNet hierarchy and its small subset [1], [147]. 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 non-experts. According to the literature [15]–[18], *four* real-world noisy datasets are widely used: ANIMAL-10N[53], real-world noisy data of human-labeled online images for 10 confusing animals [18]; Food-101N[54], real-world noisy data of crawled food images annotated by their search keywords in the Food-101 taxonomy [17], [148]; Clothing1M[55], real-world noisy data of large-scale crawled clothing images from several online shopping websites [15]; WebVision[56], real-world noisy data of large-scale web images crawled from Flickr and Google Images search [16]. 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 [21]. Hence, *test accuracy* has generally been adopted for evaluation [12]. For a test set $\mathcal{T} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{T}|}$, let $\hat{y}_i$ be the predicted label of the i-th example in $\mathcal{T}$. Subsequently, the test accuracy is formalized by

$$\text{Test Accuracy} = \frac{|\{(x_i, y_i) \in \mathcal{T} : \hat{y}_i = y_i\}|}{|\mathcal{T}|}. \quad (19)$$

If the test data are not available, *validation accuracy* can be used by replacing $\mathcal{T}$ in Eq. (19) with validation data $\mathcal{V} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{V}|}$ as an alternative,

$$\text{Validation Accuracy} = \frac{|\{(x_i, y_i) \in \mathcal{V} : \hat{y}_i = y_i\}|}{|\mathcal{V}|}. \quad (20)$$

[47] http://yann.lecun.com/exdb/mnist
[48] https://github.com/zalandoresearch/fashion-mnist
[49] https://www.cs.toronto.edu/~kriz/cifar.html
[50] http://ufldl.stanford.edu/housenumbers
[51] http://www.image-net.org
[52] https://www.kaggle.com/c/tiny-imagenet
[53] https://dm.kaist.ac.kr/datasets/animal-10n
[54] https://kuanghuei.github.io/Food-101N
[55] https://www.floydhub.com/lukasmyth/datasets/clothing1m
[56] https://data.vision.ee.ethz.ch/cvl/webvision/download.html

Furthermore, if the specified method belongs to the "sample selection" category, *label precision* and *label recall* [76], [78] can be used as the metrics,

$$\begin{aligned}\text{Label Precision} &= \frac{|\{(x_i,\tilde{y}_i)\in\mathcal{S}_t : \tilde{y}_i = y_i\}|}{|\mathcal{S}_t|},\\ \text{Label Recall} &= \frac{|\{(x_i,\tilde{y}_i)\in\mathcal{S}_t : \tilde{y}_i = y_i\}|}{|\{(x_i,\tilde{y}_i)\in\mathcal{B}_t : \tilde{y}_i = y_i\}|},\end{aligned} \tag{21}$$

where $\mathcal{S}_t$ is the set of selected clean examples in a mini-batch $\mathcal{B}_t$. The two metrics are indicators of performance for the examples selected from the mini-batch as true-labeled ones [76].

Meanwhile, if the specified method belongs to the "label refurbishment" category, *correction error* [18] can be used as an indicator of how many examples are incorrectly refurbished,

$$\text{Correction Error} = \frac{|\{x_i\in\mathcal{R} : \text{argmax}(y_i^{refurb}) \neq y_i\}|}{|\mathcal{R}|}, \tag{22}$$

where $\mathcal{R}$ is the set of examples whose labels are refurbished by Eq. (13) and y_i^{refurb} is the refurbished label of the i-th examples in $\mathcal{R}$.

VII. Future Research Directions

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

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 [63], [92], [118], [119]. However, this assumption may not be a good approximation of the real-world label noise. In particular, Chen et al. [124] conducted a theoretical hypothesis testing[56] 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. [74] pointed out that their performance could considerably worsen in the instance-dependent (or real-world) noise compared to 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 Figure 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 [83], [149] or a restricted small-scale machine learning model such as logistic regression [62]. Therefore, learning with the instance-dependent label noise is an important topic that deserves more research attention.

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

B. Multi-label Data with Label Noise

Most of the existing methods are applicable only for a *single-label* multi-class classification problem, where each data example is assumed to have only one true label. However, in the case of *multi-label* 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 [150]. In semantic scene classification, each scene may belong to multiple scene classes [151]. Thus, contrary to the single-label setup, the multi-label classifier aims to predict a set of target objects simultaneously. In this setup, a multi-label dataset of millions of examples are reported to contain over 26.6% false-positive labels as well as a significant number of omitted labels [152].

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 multi-label classification, the simple extension of existing methods may not learn the proper correlations among multiple labels. Therefore, learning from noisy labels with multi-label data is another important topic for future research. We refer the readers to a recent study [153] that discusses the evaluation of multi-label 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 simultaneously [154]. 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 [126], 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 [155]. 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 [12], [29], whereas fair training has focused on dealing with data bias, not necessarily noise [155], [156]. 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 [155]. 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 [157], [158]. 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 of adversarial training where the input feature is maliciously perturbed to distort the output of the DNN [33], [35]. 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 [159]–[161].

Interestingly, it was turned out that adversarial training makes DNNs robust to label noise [159]. Based on this finding, Damodaran et al, [160] proposed a new regularization term, called Wasserstein adversarial regularization, to address the problem of learning with noisy labels. Zhu et al. [161] 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 to traditional work. Therefore, understanding the connection between input perturbation and label noise could be another future topic for better representation learning towards 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 ther other hand, the need for more efficient algorithms is increasing owing to the rapid increase in the amount of available data [162].

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 to 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. NeurIPS*, 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, Y. Zhang, I. S. Altingovde, M. M. Rahman, P. Karagoz, A. Braylan, B. Dang, H.-L. Chang, H. Kim, Q. McNamara *et al.*, "Neural information retrieval: At the end of the early years," *Information Retrieval Journal*, vol. 21, no. 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 and Decision Making*, vol. 5, no. 5, pp. 411–419, 2010.

[10] V. Cothey, "Web-crawling reliability," *Journal of the American Society for Information Science and Technology*, vol. 55, no. 14, pp. 1228–1238, 2004.

[11] W. Mason and S. Suri, "Conducting behavioral research on amazon's mechanical turk," *Behavior Research Methods*, vol. 44, no. 1, pp. 1–23, 2012.

[12] B. Frénay and M. Verleysen, "Classification in the presence of label noise: A survey," *IEEE Transaction on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 845–869, 2013.

[13] R. V. Lloyd, L. A. Erickson, M. B. Casey, K. Y. Lam, C. M. Lohse, S. L. Asa, J. K. Chan, R. A. DeLellis, H. R. Harach, K. Kakudo *et al.*, "Observer variation in the diagnosis of follicular variant of papillary thyroid carcinoma," *The American journal of surgical pathology*, vol. 28, no. 10, pp. 1336–1340, 2004.

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

[15] 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.

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

[17] 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.

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

[19] J. Krause, B. Sapp, A. Howard, H. Zhou, A. Toshev, T. Duerig, J. Philbin, and L. Fei-Fei, "The unreasonable effectiveness of noisy data for fine-grained recognition," in *Proc. ECCV*, 2016, pp. 301–320.

[20] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio *et al.*, "A closer look at memorization in deep networks," in *Proc. ICML*, 2017, pp. 233–242.

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

[22] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019.

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

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[25] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.

[26] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study," *Artificial Intelligence Review*, vol. 22, no. 3, pp. 177–210, 2004.

[27] J. Zhang, X. Wu, and V. S. Sheng, "Learning from crowdsourced labeled data: A survey," *Artificial Intelligence Review*, vol. 46, no. 4, pp. 543–576, 2016.

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

[29] B. Han, Q. Yao, T. Liu, G. Niu, I. W. Tsang, J. T. Kwok, and M. Sugiyama, "A survey of label-noise representation learning: Past, present and future," *arXiv preprint arXiv:2011.04406*, 2020.

[30] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *Access*, vol. 6, pp. 14 410–14 430, 2018.

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

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

[33] E. Dohmatob, "Limitations of adversarial robustness: strong no free lunch theorem," in *Proc. ICML*, 2019.

[34] 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.

[35] 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*, vol. 33, no. 01, 2019, pp. 4536–4543.

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

[37] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

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

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

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

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

[42] B. Sluban, D. Gamberger, and N. Lavrač, "Ensemble-based noise detection: Noise ranking and visual performance evaluation," *Data Mining and Knowledge Discovery*, vol. 28, no. 2, pp. 265–303, 2014.

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

[44] D. Gamberger, N. Lavrac, and S. Dzeroski, "Noise detection and elimination in data preprocessing: Experiments in medical domains," *Applied Artificial Intelligence*, vol. 14, no. 2, pp. 205–223, 2000.

[45] 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.

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

[47] N. Manwani and P. Sastry, "Noise tolerance under risk minimization," *IEEE Transactions on Cybernetics*, vol. 43, no. 3, pp. 1146–1151, 2013.

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

[49] 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.

[50] 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.

[51] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, 2005.

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

[53] 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.

[54] 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.

[55] A. Ganapathiraju and J. Picone, "Support vector machines for automatic data cleanup," in *Proc. ICSLP*, 2000.

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

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

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

[59] 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.

[60] 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.

[61] 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.

[62] 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.

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

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

[65] 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. 10 456–10 465.

[66] B. C. Csáji *et al.*, "Approximation with artificial neural networks," *Faculty of Sciences, Etvs Lornd University, Hungary*, vol. 24, no. 48, p. 7, 2001.

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

[68] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Proc. NeurIPS*, 2018, pp. 8778–8788.

[69] 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.

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

[71] D. Krueger, N. Ballas, S. Jastrzebski, D. Arpit, M. S. Kanwal, T. Maharaj, E. Bengio, A. Fischer, and A. Courville, "Deep nets don't learn via memorization," in *Proc. ICLRW*, 2017.

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

[73] 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. 10 789–10 798.

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

[75] ——, "Two-phase learning for overcoming noisy labels," *arXiv preprint arXiv:2012.04337*, 2020.

[76] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *Proc. NeurIPS*, 2018, pp. 8527–8537.

[77] 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.

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

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

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

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

[82] 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.

[83] A. K. Menon, B. Van Rooyen, and N. Natarajan, "Learning from binary labels with instance-dependent noise," *Machine Learning*, vol. 107, no. 8, pp. 1561–1595, 2018.

[84] L. Torgo and J. Gama, "Regression using classification algorithms," *Intelligent Data Analysis*, vol. 1, no. 4, pp. 275–292, 1997.

[85] 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.

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

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

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

[89] 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.

[90] P. Chen, G. Chen, J. Ye, jingwei zhao, and P.-A. Heng, "Noise against noise: stochastic label noise helps combat inherent label noise," in *Proc. ICLR*, 2021.

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

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

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

[94] B. Han, J. Yao, G. Niu, M. Zhou, I. Tsang, Y. Zhang, and M. Sugiyama, "Masking: A new perspective of noisy supervision," in *Proc. NeurIPS*, 2018, pp. 5836–5846.

[95] J. Yao, J. Wang, I. W. Tsang, Y. Zhang, J. Sun, C. Zhang, and R. Zhang, "Deep learning from noisy image labels with quality embedding," *IEEE Transactions on Image Processing*, vol. 28, no. 4, pp. 1909–1922, 2018.

[96] L. Cheng, X. Zhou, L. Zhao, D. Li, H. Shang, Y. Zheng, P. Pan, and Y. Xu, "Weakly supervised learning with side information for noisy labeled images," in *Proc. ECCV*, 2020, pp. 306–321.

[97] X. Xia, T. Liu, B. Han, N. Wang, J. Deng, J. Li, and Y. Mao, "Extended T: Learning with mixed closed-set and open-set noisy labels," *arXiv preprint arXiv:2012.00932*, 2020.

[98] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. NeurIPS*, 2014, pp. 2672–2680.

[99] 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.

[100] 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. 11 244–11 253.

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

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

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

[104] X. Xia, T. Liu, B. Han, C. Gong, N. Wang, Z. Ge, and Y. Chang, "Robust early-learning: Hindering the memorization of noisy labels," in *Proc. ICLR*, 2021.

[105] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. ICLR*, 2014.

[106] G. Pereyra, G. Tucker, J. Chorowski, Ł. Kaiser, and G. Hinton, "Regularizing neural networks by penalizing confident output distributions," in *Proc. ICLRW*, 2017.

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

[108] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proc. ICLR*, 2018.

[109] 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.

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

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

[112] 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.

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

[114] R. Wang, T. Liu, and D. Tao, "Multiclass learning with partially corrupted labels," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 6, pp. 2568–2580, 2017.

[115] 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.

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

[117] X. Ma, Y. Wang, M. E. Houle, S. Zhou, S. M. Erfani, S.-T. Xia, S. Wijewickrema, and J. Bailey, "Dimensionality-driven learning with noisy labels," in *Proc. ICML*, 2018.

[118] X. Xia, T. Liu, N. Wang, B. Han, C. Gong, G. Niu, and M. Sugiyama, "Are anchor points really indispensable in label-noise learning?" in *Proc. NeurIPS*, 2019.

[119] Y. Yao, T. Liu, B. Han, M. Gong, J. Deng, G. Niu, and M. Sugiyama, "Dual T: Reducing estimation error for transition matrix in label-noise learning," in *Proc. NeurIPS*, 2020.

[120] T. Liu and D. Tao, "Classification with noisy labels by importance reweighting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 3, pp. 447–461, 2015.

[121] L. Huang, C. Zhang, and H. Zhang, "Self-adaptive training: beyond empirical risk minimization," in *Proc. NeurIPS*, 2020.

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

[123] S. Zheng, P. Wu, A. Goswami, M. Goswami, D. Metaxas, and C. Chen, "Error-bounded correction of noisy labels," in *Proc. ICML*, 2020, pp. 11 447–11 457.

[124] 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.

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

[126] J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, and D. Meng, "Meta-Weight-Net: Learning an explicit mapping for sample weighting," in *Proc. NeurIPS*, 2019, pp. 1917–1928.
[127] Z. Wang, G. Hu, and Q. Hu, "Training noise-robust deep neural networks via meta-learning," in *Proc. CVPR*, 2020, pp. 4524–4533.
[128] M. Dehghani, A. Severyn, S. Rothe, and J. Kamps, "Learning to learn from weak supervision by full supervision," in *Proc. NeurIPSW*, 2017.
[129] ——, "Avoiding your teacher's mistakes: Training neural networks with controlled weak supervision," *arXiv preprint arXiv:1711.00313*, 2017.
[130] 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.
[131] 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.
[132] G. Zheng, A. H. Awadallah, and S. Dumais, "Meta label correction for noisy label learning," in *Proc. AAAI*, 2021.
[133] 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.
[134] Y. Wang, W. Liu, X. Ma, J. Bailey, H. Zha, L. Song, and S.-T. Xia, "Iterative learning with open-set noisy labels," in *Proc. CVPR*, 2018, pp. 8688–8696.
[135] 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.
[136] J. Huang, L. Qu, R. Jia, and B. Zhao, "O2U-Net: A simple noisy label detection approach for deep neural networks," in *Proc. ICCV*, 2019, pp. 3326–3334.
[137] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," *ACM SIGMOD Record*, vol. 29, no. 2, pp. 93–104, 2000.
[138] P. Wu, S. Zheng, M. Goswami, D. Metaxas, and C. Chen, "A topological filter for learning with label noise," in *Proc. NeurIPS*, 2020.
[139] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proc. NeurIPS*, 2017, pp. 1195–1204.
[140] 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.
[141] T. Zhou, S. Wang, and J. Bilmes, "Robust curriculum learning: from clean label detection to noisy label self-correction," in *Proc. ICLR*, 2021.
[142] G. Pleiss, T. Zhang, E. R. Elenberg, and K. Q. Weinberger, "Detecting noisy training data with loss curves," 2020. [Online]. Available: https://openreview.net/forum?id=HyenUkrtDB
[143] Y. LeCun, C. Cortes, and C. J. Burges, "The MNIST database of handwritten digits, 1998," *URL http://yann. lecun. com/exdb/mnist*, vol. 10, p. 34, 1998.
[144] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.
[145] A. Krizhevsky, V. Nair, and G. Hinton, "CIFAR-10 and CIFAR-100 datasets," 2014, https://www.cs.toronto.edu/~kriz/cifar.html.
[146] 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.
[147] A. Karpathy *et al.*, "Cs231n convolutional neural networks for visual recognition," *Neural Networks*, vol. 1, p. 1, 2016.
[148] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101–mining discriminative components with random forests," in *Proc. ECCV*, 2014, pp. 446–461.
[149] J. Bootkrajang and J. Chaijaruwanich, "Towards instance-dependent label noise-tolerant classification: a probabilistic approach," *Pattern Analysis and Applications*, vol. 23, no. 1, pp. 95–111, 2020.
[150] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *International Journal of Data Warehousing and Mining*, vol. 3, no. 3, pp. 1–13, 2007.
[151] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.
[152] I. Krasin, T. Duerig, N. Alldrin, V. Ferrari, S. Abu-El-Haija, A. Kuznetsova, H. Rom, J. Uijlings, S. Popov, A. Veit *et al.*, "OpenImages: A public dataset for large-scale multi-label and multi-class image classification," *Dataset available from https://github. com/openimages*, vol. 2, no. 3, p. 18, 2017.
[153] W. Zhao and C. Gomes, "Evaluating multi-label classifiers with noisy labels," *arXiv preprint arXiv:2102.08427*, 2021.
[154] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, pp. 1–54, 2019.
[155] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," in *Proc. NeurIPS*, 2016.
[156] H. Jiang and O. Nachum, "Identifying and correcting label bias in machine learning," in *Proc. AISTATS*, 2020, pp. 702–712.
[157] 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.
[158] J. Wang, Y. Liu, and C. Levy, "Fair classification with group-dependent label noise," in *Proc. FAccT*, 2021, pp. 526–536.
[159] J. Uesato, J.-B. Alayrac, P.-S. Huang, R. Stanforth, A. Fawzi, and P. Kohli, "Are labels required for improving adversarial robustness?" in *NeurIPS*, 2019.
[160] 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.
[161] J. Zhu, J. Zhang, B. Han, T. Liu, G. Niu, H. Yang, M. Kankanhalli, and M. Sugiyama, "Understanding the interaction of adversarial training with noisy labels," *arXiv preprint arXiv:2102.03482*, 2021.
[162] G. Nguyen, S. Dlugolinsky, M. Bobák, V. Tran, Á. L. García, I. Heredia, P. Malík, and L. Hluchỳ, "Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 77–124, 2019.