Trustable Co-label Learning from Multiple Noisy Annotators

Shikun Li, Tongliang Liu, Senior Member, IEEE, Jiyong Tan, Dan Zeng, Senior Member, IEEE and Shiming Ge, Senior Member, IEEE

Abstract—Supervised deep learning depends on massive accurately annotated examples, which is usually impractical in many real-world scenarios. A typical alternative is learning from multiple noisy annotators. Numerical earlier works assume that all labels are noisy, while it is usually the case that a few trusted samples with clean labels are available. This raises the following important question: how can we effectively use a small amount of trusted data to facilitate robust classifier learning from multiple annotators? This paper proposes a data-efficient approach, called Trustable Co-label Learning (TCL), to learn deep classifiers from multiple noisy annotators when a small set of trusted data is available. This approach follows the coupled-view learning manner, which jointly learns the data classifier and the label aggregator. It effectively uses trusted data as a guide to generate trustable soft labels (termed co-labels). A co-label learning can then be performed by alternately reannotating the pseudo labels and refining the classifiers. In addition, we further improve TCL for a special complete data case, where each instance is labeled by all annotators and the label aggregator is represented by multilayer neural networks to enhance model capacity. Extensive experiments on synthetic and real datasets clearly demonstrate the effectiveness and robustness of the proposed approach. Source code is available at https://github.com/ShikunLi/TCL.

Index Terms—Label noise, multiple annotators, crowdsourcing, learning from crowds

I. INTRODUCTION

RECENT deep learning has achieved state-of-the-art results in various classification tasks, e.g., image recognition [1], object detection [2] and text matching [3]. These advances are largely due to the availability of large-scale cleanly annotated datasets and effective model learning algorithms. However, massive clean annotations are very difficult to collect in many real-world scenarios, e.g., video surveillance in the wild [4], medical data analysis [5] and webly supervised multimedia understanding [6]–[10]. Noisy labels are usually easier to acquire in these scenarios. Since the capacity of deep networks is so high that they can totally memorize noisy labels [11], designing deep methods robust to noisy labels is challenging [12]–[14].

S. Li and S. Ge are with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100095, China, and with School of Cyber Security at University of Chinese Academy of Sciences, Beijing 100049, China. Email: {lishikun,geshiming}@iie.ac.cn
T. Liu is with the Trustworthy Machine Learning Lab, the University of Sydney, 6 Cleveland St, Darlington, NSW 2008, Australia. Email: tongliang.liu@sydney.edu.au
J. Tan is with the AISONO AIR Lab, Shenzhen, and with the Harbin Institute of Technology. Email: scutjy2015@163.com
D. Zeng is with the Department of Communication Engineering, Shanghai University, Shanghai 200040, China. E-mail: dzeng@shu.edu.cn
S. Ge is the corresponding author. E-mail: geshiming@iie.ac.cn

Typically, learning from multiple noisy annotators (e.g., different non-expert persons, weak models, weak discriminant rules, or other automatic labeling sources) provides an alternative way to lessen label noise [15]–[17]. The common practice is to aggregate multiple weak labels for one instance to obtain a more reliable label and then learn with the aggregated labels. Following this line, many label aggregation methods have been proposed, e.g., majority vote [18], participant-mine voting [19], Dawid-Skene estimator [20] and Bayesian classifier combination [21]. In addition, a promising direction is to introduce the instance feature to help decrease the influence of label noise [22]–[26]. Recently, some studies aimed in this direction jointly learn a deep classifier and a label aggregator [27]–[29], which provides a flexible and general coupled-view deep learning manner to address the issue. However, these works assume that all labels are noisy, while it is usually the case that a few trusted examples with clean labels are available. This means that if we effectively introduce such trusted data into training, it will enable substantial label corruption robustness performance gains. Then, this raises the following important question: how can we effectively use the trusted data to facilitate robust classifier learning from multiple annotators?

An intuitive way to utilize the small, trusted dataset is fine-tuning the pretrained models [30], where the trusted data usually do not play a role in the pretraining process and the small amount of data limits its performance gains. Thus, we propose to improve the model performance during the whole learning process with the help of the small, trusted dataset. As illustrated in Fig. 1, our idea is to jointly model the data classifier by deep neural networks and the label aggregator by a Naive Bayes classifier in a coupled-view learning fashion. During training, our approach alternately refines classifiers and combines the predictions into trustable soft labels (termed...
co-labels) with the guidance of the small, trusted dataset. In the iterative learning process, to achieve a trustable prediction combination, we first utilize the trusted data to calibrate the data classifier, which makes the confidences from the deep classifier reflect the ground truth correctness likelihood \[31\]. Then, based on the conditionally independent probability assumption, we combine the label probabilities from both classifiers into a joint probability, which is regarded as a trustable soft label distribution for robust classifier learning in the next iteration. After the alternate optimization, our approach reinitializes the data classifier, which is then retrained on the union of reannotated untrusted and trusted data. In this way, the small, trusted dataset provides both its original clean target for classifier learning and extra guidance for the reliable labeling of untrusted data, leading to data-efficient learning. Our approach is named Trustable Co-label Learning (TCL) since it effectively makes use of a small, trusted dataset to progressively obtain more trustable labels by combining the data classifier and the label aggregator.

In addition, we design a variant method of TCL for a special complete data case, where all instances are labeled by every annotator. Note that this kind of data can be collected from various automatic labeling sources, e.g., weak classifiers \[32\], user behaviors \[33\] and search engines \[34\]. To achieve reliable co-label updates, we similarly obtain well-calibrated deep networks and combine predictions via the trusted data. Different from the original TCL, we model both the data classifier and the label aggregator by multilayer neural networks. It is natural that the performance of label aggregator benefits from the high modeling capacity of deep networks.

To verify the effectiveness and robustness of the proposed approach, we conduct experiments on synthetic and real datasets under two learning settings, i.e., learning from non-expert humans (sparse data case\[1\]) and learning from auto-labeling sources (complete data case). The results clearly show that our approach can substantially combat the negative impact of label noise by using a small, trusted dataset effectively.

The main contributions of this paper are summarized in three aspects: 1) We propose a data-efficient approach, called Trustable Co-label Learning (TCL), to learn a robust classifier from multiple noisy annotators when a small, trusted dataset is available. It effectively uses trusted data to combine the predicted distributions into the trustable label distributions. 2) We further improve TCL for a special complete data case, where all instances are labeled by every annotator, and the label aggregator is represented by multilayer neural networks to enhance modeling ability. 3) We conduct experiments on synthetic and real datasets, which clearly demonstrate that our approach outperforms state-of-the-art approaches in terms of effectiveness and robustness.

II. RELATED WORKS

We briefly review the related works from three aspects in this section, including learning with multiple noisy labels, multi-view methods and hybrid methods.

In this paper, the sparse data case means each annotator only labels part of training data, and the complete data case means all training samples are labeled by every annotator.

A. Learning with Multiple Noisy Labels

When multiple noisy annotators are available for each instance, one basic direction is to infer true labels from multiple noisy labels and then learn a data classifier with those inferred labels. Probabilistic generative methods and discriminative methods are mainly two strands to address such label aggregation problems. Generative methods generally build a probabilistic model to generate noisy observations conditioned on unknown true labels and some behavior assumptions, e.g., the Dawid-Skene estimator \[20\], the minimax entropy estimator \[35\]. Bayesian classifier combination \[21\], structure learning \[40\] and their variants \[18\], \[37\]–\[41\]. In contrast, discriminative methods do not model the observations but directly identify the true labels via aggregation rules. Between them, the simplest but effective method is majority voting \[42\], which Naively assumes that all annotators are equally reliable. Advanced approaches take different reliabilities of workers or instances into consideration, including weighted majority voting \[19\], \[43\], graph modeling \[44\], max-margin majority voting \[45\], tensor factorization methods \[46\]–\[48\], etc.

Recently, a promising direction has been to introduce instance features to help decrease the influence of label noise, especially to jointly learn data classifiers. Many works in this direction propose an Expectation Maximization (EM) algorithm for jointly learning the levels of expertise of different annotators and the parameters of a classifier \[22\], \[24\], \[25\]. Rodrigues et al. \[49\] propose adding a crowd layer to the output of a common network, and the layer adjusts the gradients coming from the labels of annotators. DoctorNet \[50\] learns different models for every annotator, and the whole output is weighted integration of multiple models’ predictions. Chu et al. \[51\] also add a crowd layer but decompose the confusion matrix into a commonly shared confusion matrix and an individual confusion matrix. Cao et al. \[27\] provide an information theoretic method that interprets the joint learning from multiple-annotators problem as a coupled-view problem. Li et al. \[29\] introduce a coupled-view method with several robust learning schemes to address it. Generally, these works assume that all labels are noisy and may be suboptimal when a small, trusted dataset is available. Therefore, we effectively use the trusted data in a coupled-view learning manner to improve the learning performance.

B. Multi-view Methods

Multi-view methods mainly include co-training style algorithms, co-regularization style algorithms, multi-kernel learning algorithms and other types. Co-training style algorithms \[52\]–\[54\] train learners alternately on distinct views with confident labels for unlabeled data. Coregularization style algorithms \[55\], \[56\] regard disagreement between multiple views as a regularization term in the objective function. Multi-kernel learning algorithms \[57\], \[58\] exploit kernels that naturally correspond to different views and combine kernels to improve learning performance. In addition, multi-view graph clustering \[59\] and multi-view subspace clustering \[60\] also belong to multi-view learning.
Jointly learning a data classifier and a label aggregator can be seen as a coupled-view learning problem, but the above methods are mainly applied to unsupervised and semi-supervised learning, which cannot be directly applied to noisy supervised learning. This paper proposes trustable co-label learning in the learning-from-multiple-noisy-annotators setting, where co-labels act as the information exchange bridge between two classifiers of different views.

C. Hybrid Methods

Given the output of a set of base classifiers, rather than trying to find the best single learner, hybrid methods (also called combination methods) aim to integrate learners to enhance the generalization ability.

For nominal outputs, majority voting is the most common hybrid method [61]. Weighted voting [62] assigns more weight to stronger classifiers for voting. The Naive Bayes combiner [63] assumes that the classifiers are mutually independent given a true class label and takes the maximum label for the posterior probability. Other combination methods for such outputs include the BKS method [64], Wernecke method [65], SVD combination method [66], Bayesian classifier combination [21], etc. For numeric outputs, combiners can be classified into two main categories: nontrainable and trainable methods. Nontrainable combiner [67] has no extra parameters that need to be trained, including simple averaging, minimum/maximum/median rule, product rule, and generalized mean [68]. Trainable combiners include weighted averaging [69]-[71], fuzzy integral [72], stacking [73], decision template method [74], selection methods [75], etc. Note that integrating classifiers can be regarded as a special form of the label aggregation problem, and some crowdsourcing methods can also be used for it.

Generally, there is no unique best combiner for all problems. In this paper, we are faced with two hybrid problems, i.e., how to integrate learners of different views. For the first problem, our approach learns a label combiner modeled by a Naive Bayes classifier or multilayer neural networks. For the second one, we combine the label probabilities from both classifiers into a joint probability under the guidance of a small, trusted dataset.

III. OUR APPROACH

A. Problem Formulation

Preliminaries. We are given a massive untrusted training dataset \( D \) of \( n \) instances drawn from \( p(X, Y) \). It includes data features \( x = \{x_i\}_{i=1}^n \) and multiple noisy labels \( \tilde{y} = \{y_i\}_{i=1}^n \), where \( y_i = (y^{(1)}_i, y^{(2)}_i, ..., y^{(m)}_i) \) is an \( m \)-dimensional vector from fixed \( m \) labeling sources (it may have missing labels). \( C \) is the size of label space, and \( y^{(m)}_i \in \{1, 2, ..., C\} \).

We are given a small, trusted training dataset \( D \) of \( u \) examples drawn from \( p(X, Y) \), which includes data features and corresponding clean labels.

Overview. Following coupled-view learning, our approach needs to model two classifiers: 1) label aggregator \( f_\ell(\tilde{y}; w_\ell) \), which combines an \( m \)-dimensional weak label vector \( \tilde{y} \) from fixed \( m \) annotators to produce an estimated distribution, and 2) data classifier \( f_d(x; w_d) \), which is a deep classifier that takes an instance feature \( x \) as the input and outputs a predicted distribution. Here, \( w_d \) and \( w_\ell \) are the learned parameters.

To achieve effective and robust learning, our approach alternately conducts a classifier learning phase and prediction combination phase under the guidance of a small, trusted dataset (see Fig. 2a), and the performance of the two training phases is progressive. After that, it performs retraining on untrusted data with fixed co-labels and trusted data with given clean labels to attain the final data classifier. Next, we introduce the two iterative phases in detail.

B. Classifier Learning

With the help of co-labels \( y^c = \{y^c_i\}_{i=1}^n \) that are combined from the predictions of \( f_\ell \) and \( f_d \) during training, we treat this weakly supervised learning problem as a supervised learning problem in the classifier learning phase:

\[
\min_{w_d, w_\ell} \ell_d(x, y^c; w_d) + \ell_l(\tilde{Y}, y^c; w_\ell), \tag{1}
\]

where \( \ell_d(\cdot) \) and \( \ell_l(\cdot) \) are the loss functions for training \( f_d \) and \( f_\ell \), respectively.

Data classifier. Due to its high capacity to learn from data, we regard a deep network \( f_d(x; w_d) \) as the data classifier. With the supervision of soft co-labels, its loss function can be written as Eq. (2):

\[
\ell_d(x, y^c; w_d) = \sum_{i=1}^n \ell_{CE}(f_d(x_i; w_d), y^c_i), \tag{2}
\]

where \( \ell_{CE}(\cdot) \) is the cross-entropy function and \( y^c_i \) is the \( C \)-dimensional co-label vector of instance \( i \).

Label aggregator. The label aggregator aims to infer an estimated label distribution for each instance from \( m \) labeling sources. Similar to Li et al. [29], with the assumption that every annotator is conditionally independent given true labels, we adopt a Naive Bayes classifier to model label aggregation. When the confusion matrices \( \pi \) and the class prior probability \( q \) are known, \( f_\ell(\tilde{y}_i; w_\ell = (\pi, q)) = K(\tilde{Y} = \tilde{y}_i; \pi, q) \), the posterior probability of the true label of instance \( i \) for class \( k \) can be calculated as Eq. (3):

\[
f_\ell(\tilde{y}_i; \pi, q)_k = \frac{q_k \prod_{j=1}^m (\sum_{s=1}^C I[\tilde{y}_i^{(j)} = s] \pi_k^{(j)})}{\sum_{k'=1}^C (q_{k'} \prod_{j=1}^m (\sum_{s=1}^C I[\tilde{y}_i^{(j)} = s] \pi_{k'}^{(j)})}, \tag{3}
\]

where \( \tilde{y}_i^{(j)} \) is the noisy label of instance \( i \) from annotator \( j \); \( I[\cdot] \) is the indicator function, which takes 1 if the event is true and 0 otherwise; \( \pi_{k'}^{(j)} \) is the probability of misclassifying class \( k \) into class \( s \) for annotator \( j \); and \( q_k \) is the prior probability of class \( k \).

With the supervision of soft co-labels, its loss function can be written as:

\[
\ell_l(\tilde{Y}, y^c; \pi, q) = \sum_{i=1}^n \ell_{NLL}(f_\ell(\tilde{y}_i; \pi, q), y^c_i), \tag{4}
\]

where \( \ell_{NLL}(\cdot) \) is a negative log-likelihood function and \( y^c_i \) is the \( C \)-dimensional co-label vector of instance \( i \).
To solve this problem, \( \pi \) is acquired by Eq. (5), and \( q \) is estimated from trusted data \( D \):

\[
\pi^{(j)}_{ks} = \frac{\sum_{i=1}^{n} I[y_i^{(j)} = s]y_{ik}^c}{\sum_{i=1}^{n} y_{ik}^c},
\]

where \( y_{ik}^c \) is the co-label for class \( k \) of instance \( i \).

### C. Prediction Combination

As the simple intuition illustrated in Fig. 1, we want to make use of the trusted data to combine the predictions \( \phi_d(x_i) \) and \( \phi_i(y_i) \) into more trustable co-labels \( y_i^c \) in each training iteration to make the co-label learning efficient and robust. This practice agrees with the observations in recent works \[76\], \[77\] that the prediction from deep networks during training contains information about the true label distribution, which can be used to reduce label noise through label correction \[28\], \[78\] or sample selection \[79\]–\[81\].

Our prediction combination process is based on the perspective of probability. In classical machine learning, soft prediction from classifiers can be seen as an estimation of the posterior probability. However, as modern neural networks are usually poorly calibrated \[51\], \( \phi_d(x_i) \), the prediction from deep networks, cannot be effectively representative of the probability, which makes the confidences from the data classifier and label aggregator hard to compare and combine directly. This is where a small, trusted dataset helps, i.e., we use it to calibrate the prediction from deep neural networks for combination. In the experiments, we use the isotonic regression method \[82\] to perform calibration. Fig. 3 clearly shows the important effect of calibration. As reported, the VGG-16 model learned with initialized co-labels on the IND-3 dataset is very poorly calibrated, which means that its Calibration Error (ECE) of the data classifier decreases from 9.87 to 3.67, and its accuracy even increases from 30.09% to 43.29%. Subsequent experiments demonstrate that such calibration via trusted data plays a key role in guiding trustable prediction combination and further encourages robust classifier learning.

Therefore, with trusted data \( D \), we calibrate \( \phi_d(x_i) \) to let it estimate the true label likelihood, which means we assume the well-calibrated prediction as \( \hat{P}(Y|S = s_i, D) \), where \( s_i = \phi_d(x_i) \). In addition, \( \phi_i(y_i) \) the prediction from Naive Bayes classifier can be seen as the estimated distribution \( \hat{P}(Y|\tilde{Y} = y_i) \).

With the well-estimated distributions, we can combine two predictions into a joint distribution based on the conditionally independent assumption, which means that assuming \( \tilde{Y} \) and \( \{S, D\} \) are independent given true label \( Y \), \( P(Y|S, D, \tilde{Y}) \) can be inferred by Eq. (6):

\[
P(Y|S, D, \tilde{Y}) = \frac{P(Y)p(S, D, \tilde{Y}|Y)}{p(S, D, \tilde{Y})} = \frac{P(Y)p(S, D|Y)p(\tilde{Y}|Y)}{p(S, D, \tilde{Y})} = \frac{P(Y)p(Y|S, D)p(S, D)P(\tilde{Y}|Y)P(Y)}{p(S, D, \tilde{Y})} = \frac{P(Y)p(\tilde{Y}|Y)P(Y|S, D)p(S, D)}{p(S, D, \tilde{Y})} = \frac{P(Y)p(Y|S, D)p(\tilde{Y}|Y)P(\tilde{Y}|Y)}{P(Y)}.
\]

As \( P(Y|S, D, \tilde{Y}) \propto P(Y|S, D)p(\tilde{Y}|Y)/P(Y) \), we acquire
the combined distribution by Eq. (7):
\[ \hat{P}(Y|s_i, D, \tilde{y}_i) = \text{Nor}(\hat{P}(Y = k|s_i, D) \cdot \hat{P}(Y = k|\tilde{y}_i)) \]

where \( \text{Nor}[r] := \frac{r}{\sum_{k=1}^n r_k} \).

After that, in each prediction combination phase, the co-labels are updated by the combined distributions. Note that although the independent assumption is not satisfied in real situations, the soft distributions are empirically verified to be more reliable than classifiers in our experiments.

D. Alternate Optimization

The whole approach is shown in Alg. 1. After the co-labels are initialized by majority voting, alternate optimization of the two training phases is conducted. During training iteration, reliable combined distributions make classifier learning more robust, and then better classifiers encourage more reliable combination in the next iteration. By such alternate optimization, our approach progressively attains more trustable co-labels.

In addition, as the deep classifier may overfit noise when the co-labels are highly noisy in the first few rounds, to mitigate this impact and further utilize trusted data, after the alternate optimization stage, our approach reinitializes the data classifier and performs the retraining stage, which learns deep networks on untrusted data with fixed co-labels and trusted data with given clean labels. In this way, the small, trusted dataset not only provides the original clean target for learning but also serves as a guide to alleviate the unreliability of co-labels.

E. Our Approach for A Special Complete Data Case

Typically, when instances are labeled by multiple non-expert human workers, the label data \( \tilde{y} \) are very sparse, which means that each worker annotates only a part of the samples, leading to many missing labels. This property limits the modeling method of label aggregation since some powerful modeling methods, such as multilayer neural networks, cannot handle the serious missing problem well. However, recently, with the huge need for large-scale labeled datasets, the deep learning community has a great interest in making use of automatic labeling methods, such as weak classifiers from small-scale data [9], [32] or similar domains [83], messy user tags from social media [33], [84], matched results from search engines [34], [85], relations from knowledge bases [8] and other human-free labeling sources. The good news is that it is easier to label all instances without missing by those automatic labeling sources in some scenarios (e.g., social network tagging [86], [87] and financial analysis [88]). This motivates us to improve our approach (see Fig. 2b) by modeling the label aggregator as a multilayer neural network for a special complete data case where every annotator labels all training data. For simplicity, we introduce the differences of the improved approach (denoted as TCLS) from the original TCL as follows:

Training dataset. We are given a massive untrusted dataset \( D = \{x, \tilde{y}\} \), where \( \tilde{y} \) is complete (i.e., it has no missing labels), and a small trusted dataset \( D \) drawn from \( p(X, Y, Y) \), which includes data features, complete multiple noisy labels, and clean labels.

Label aggregator learning. To enhance modeling ability, the label aggregator \( \phi_i \) is represented by multilayer neural networks. Its loss function can be written as:
\[ \ell_i(\tilde{y}, y^\ast; w_i) = \sum_{i=1}^{\text{CE}} (\phi_i(\tilde{y}_i; w_i), y_i^\ast) \] (8)

Prediction combination. Following the same inspiration as TCL, we reannotate co-labels by effectively utilizing trusted data. First, we calibrated \( \phi_d(x_i) \) and \( \phi_i(\tilde{y}_i) \) via the trusted dataset \( D \), and then regard them as \( \hat{P}(Y|S = s_i, D) \) and \( \hat{P}(Y|V = v_i, D) \), respectively, where \( s_i = \phi_d(x_i) \) and \( v_i = \phi_i(\tilde{y}_i) \). Assuming \( V \) and \( S \) are conditionally independent given \( Y \), \( p(S, D|Y) \approx p(D)p(S|Y) \) and \( p(V, D|Y) \approx p(D)p(V|Y) \), we can obtain \( P(Y|S, D, V) \propto P(Y|S, D)p(Y|V, D)/P(Y) \); thus, we update co-labels by Eq. (9):
\[ \hat{P}(Y|s_i, D, v_i) = \text{Nor}(\hat{P}(Y = k|s_i, D) \cdot \hat{P}(Y = k|v_i, D)) \] (9)

In addition, to utilize the existing knowledge, we initialize co-labels from the predictions of Naive Bayes classifier, which is trained on \( D \).

Algorithm 1: Trustable Co-label Learning from Multiple Noisy Annotators

Input: untrusted dataset \( \tilde{D} = \{x, \tilde{y}\} \), trusted dataset \( D \); max iteration round \( T \).
Output: learned parameters \( w_d \) of deep classifier \( \phi_d \).
1: Initialize co-labels \( y^\ast \) by majority voting from \( \tilde{y} \) and initialize the parameters \( w_d \) of \( \phi_d \).
2: for \( t = 1, 2, ..., T \) do
3: \hspace{10pt} Train \( \phi_i \) on \( \{\tilde{y}, y^\ast\} \).
4: \hspace{10pt} Combine predictions and update \( y^\ast \) by Eq. (7).
5: \hspace{10pt} Train \( \phi_d \) on \( \{x, y^\ast\} \).
6: \hspace{10pt} Combine predictions and update \( y^\ast \) by Eq. (7).
7: end for
8: Reinitialize \( \phi_d \) and retrain it on \( \{x, y^\ast\} \) and \( D \).
9: return \( w_d \).

Algorithm 2: Trustable Co-label Learning for a Special Complete Data Case

Input: untrusted dataset \( \tilde{D} = \{x, \tilde{y}\} \), trusted dataset \( D \); max iteration round \( T \).
Output: learned parameters \( w_d \) of deep classifier \( \phi_d \).
1: Initialize co-labels \( y^\ast \) from the predictions of Naive Bayes classifier, which is trained on trusted data \( D \), and initialize the parameters \( w_d \) of \( \phi_d \) and \( w_i \) of \( \phi_i \).
2: for \( t = 1, 2, ..., T \) do
3: \hspace{10pt} Train \( \phi_i \) on \( \{\tilde{y}, y^\ast\} \).
4: \hspace{10pt} Combine predictions and update \( y^\ast \) by Eq. (9).
5: \hspace{10pt} Train \( \phi_d \) on \( \{x, y^\ast\} \).
6: \hspace{10pt} Combine predictions and update \( y^\ast \) by Eq. (9).
7: end for
8: Reinitialize \( \phi_d \) and retrain it on \( \{x, y^\ast\} \) and \( D \).
9: return \( w_d \).
With the high modeling capacity of deep networks, it is natural that the performance of label aggregator benefits from it, and we verify this in our experiments. The algorithm process is shown in Alg. 2.

IV. EXPERIMENTS

To verify the effectiveness and robustness of our approach, we conduct experiments on both synthetic and real datasets under two settings: learning from non-expert humans and learning from auto-labeling sources.

A. Learning from Non-expert Humans

Learning from non-expert humans obtains data labeled by more than one person. It is a typical and well-known setting for learning from multiple noisy annotators, and the label data are usually sparse. In this section, we conduct experiments under such setting on one synthetic dataset (CIFAR10 [89]) and two real datasets (LabelMe-AMT [49] and CUBShape [90]) to evaluate our TCL.

Datasets. CIFAR10 is a 10-class image classification dataset that consists of 50K training images and 10K validation images. We retain 1K samples (each class has 100 samples) of the training data for trusted data and corrupt the other data manually by the confusion matrix $Q$, where $Q_{ij} = P(Y = j|Y = i)$, given that one noisy label $Y$ is flipped from clean label $Y$. We produce noisy labels with three kinds of confusion matrices, including 1) symmetry flipping, which simulates that the annotator may choose false labels uniformly at random with probability $\varepsilon$; 2) pair flipping, which imitates the annotator who may confuse similar classes with probability $\varepsilon$; and 3) class-wise flipping, which simulates that the annotator only does good labeling in particular classes but chooses labels uniformly at random for other classes. In our problem setting, the untrusted dataset is labeled by 30 noisy annotators, and each instance has 3 weak labels from three randomly chosen different annotators. All annotators are produced from 3 different confusion matrices $Q$, i.e., one matrix produces 10 annotators. To cover more cases, we design 4 sets of independent annotators in such a setting, resulting in 4 noisy datasets (see Tab. 1). For training, we pad the pictures on all sides by 4 pixels, randomly crop them by the size of $32 \times 32$, apply random horizontal flip, and finally normalize them. For validation, we only normalize the pictures.

LabelMe-AMT is a real-world 8-class image classification dataset. It consists of a total of 2,688 images, where 1,000 of them are used to obtain noisy labels by an average of 2.5 workers per image (59 workers in total) from Amazon Mechanical Turk. 80 images (each class 10 samples) are used for trusted data, while 1608 images are used for validation. We follow the image preprocessing method in [49].

The real dataset CUBShape is adapted from the shape task (the shape is perch-like or not) of the CUB-200-2010 dataset [90]. It contains the binary labeling task to label the shape for 6,033 bird images from Amazon Mechanical Turk. There are approximately 500 users contributing labels, and each image receives 5 labels. We collect ground truth from whatbird.com for evaluation. We retain 5000 images for untrusted data, 100 images for trusted data, and 933 images for validation data. For training, we apply random horizontal flip, cropping ($448 \times 448$) and cutout and finally normalize them. For validation, we only apply center cropping and normalization.

Implementation. For synthetic CIFAR10 datasets, we adopt the VGG-16 networks [91] to model the data classifier. Our TCL approach trains two classifiers for $T = 60$ iterations and then retraining the data classifier for 60 epochs. In each iteration, the data classifier is trained for 1 epoch. During the alternate optimization stage and retraining stage, we use SGD with a batch size of 128, a momentum of 0.9, a weight decay of 0.0005, and an initial learning rate of 0.1. The learning rate is divided by 10 after 40 epochs and 50 epochs (for a total of 60 epochs).

For the real LabelMe-AMT dataset, we use the pretrained CNN layers of the VGG-16 network and apply only one FC layer (with 128 units and ReLU activations) and one output layer on top with 50% dropout. Our TCL approach trains two classifiers for $T = 29$ iterations, and then retrained the data classifier for 30 epochs. In the first iteration round, the data classifier is trained for 2 epochs, and in the other round for 1 epoch (30 epochs in total). We used the Adam optimizer with a batch size of 128, a learning rate of 0.0001, and betas of (0.9, 0.999).

For real the CUBShape dataset, we also use the pretrained CNN layers of the VGG-16 network and apply only one FC layer (with 128 units and ReLU activations) and one output layer on top with 50% dropout. Our TCL approach trains two classifiers for $T = 29$ iterations, and then retrained the data classifier for 10 epochs. In the first iteration round, the data classifier is trained for 2 epochs, and in the other round, it is trained for 1 epoch (30 epochs in total). During the alternate optimization stage and retraining stage, we use SGD with a batch size of 16, a momentum of 0.9, a weight decay of 0.0005, and an initial learning rate of 0.01. The learning rate is divided by 10 after 20 epochs.

Results. First, we compare our TCL approach with several label aggregation methods without using data features to train the deep neural network, whose estimated probabilities are used directly as learning targets (as done in knowledge distillation [92]). Including (i) DL-MV that trains a DNN on the result of (hard) majority voting [42]; (ii) DL-DS that trains a DNN on the result of the Dawid-Skene estimator [20]; (iii) DL-IBCC that trains a DNN on the result of the independent Bayesian classifier combination [21]; and (iv) DL-CRH that trains a DNN on the result of CRH model [19]. Second, we compare it with the approaches that combine data classifier learning and annotator modeling in a joint manner, including (i) AggNet [24] that uses EM algorithm to jointly estimate workers’ skills and a data classifier; (ii) Crowd Layer [49] that adds a crowd layer to the output of a common network to model confusion matrices; (iii) MBEM [28] is an improved EM algorithm that rewrites the EM likelihood and regards the estimated true labels as hard labels; (iv) Max-MIG [27] is

\footnote{Since the original version of IBCC [21] using Gibbs sampling is expensive to run and therefore unusable when thousands of labels are present, we use its variant using variational Bayes [38].}
TABLE I: The generated datasets by simulating different independent annotators on CIFAR10.

| Datasets      | 1st group of annotators |
|---------------|-------------------------|
| IND-1        | Symmetry, $\varepsilon = 0.8$ |
| IND-2        | Symmetry, $\varepsilon = 0.85$ |
| IND-3        | Symmetry, $\varepsilon = 0.8$ |
| IND-4        | Symmetry, $\varepsilon = 0.6$ |

| 2nd group of annotators |
|-------------------------|
| Symmetry, $\varepsilon = 0.7$ |
| Pair, $\varepsilon = 0.45$ |
| Symmetry, $\varepsilon = 0.7$ |
| Classwise, correct class 1 |
| Classwise, correct class 1,2,3 |
| Classwise, correct class 1,2,3,4 |

| 3rd group of annotators |
|-------------------------|
| Symmetry, $\varepsilon = 0.45$ |
| Pair, $\varepsilon = 0.45$ |
| Symmetry, $\varepsilon = 0.7$ |
| Classwise, correct class 1,2,3,4 |

TABLE II: Validation accuracy (%) on synthetic CIFAR10 datasets with independent annotators. "F" denotes fine-tuning on trusted data. The minimal improvement is also given.

| Approach                | IND-1 | IND-2 | IND-3 | IND-4 |
|-------------------------|-------|-------|-------|-------|
| DL-MV                   | 71.77 | 52.19 | 44.72 | 68.94 |
| DL-DS                   | 89.88 | 80.89 | 62.03 | 83.39 |
| DL-IBC                 | 90.43 | 77.05 | 50.42 | 84.08 |
| DL-CRH                 | 86.80 | 88.23 | 37.68 | 76.82 |
| AggNet                  | 91.42 | 88.79 | 82.23 | 88.81 |
| CrowdLayer               | 89.32 | 88.42 | 80.27 | 83.96 |
| MBEM                   | 90.90 | 89.77 | 79.63 | 87.64 |
| Max-MIG                 | 90.29 | 88.59 | 83.47 | 88.00 |
| CVL                    | 88.72 | 86.93 | 76.12 | 84.37 |
| DL-MV+F                  | 81.68 | 72.68 | 61.06 | 80.02 |
| DL-DS+F                  | 89.93 | 82.72 | 64.03 | 83.82 |
| DL-IBC+F                | 90.46 | 82.40 | 64.97 | 84.78 |
| DL-CRH+F                | 90.36 | 90.43 | 39.81 | 77.12 |
| AggNet+F              | 91.56 | 89.57 | 82.33 | 88.90 |
| CrowdLayer+F          | 89.60 | 90.64 | 80.57 | 83.99 |
| MBEM+F              | 90.96 | 90.66 | 80.68 | 87.84 |
| Max-MIG+F            | 90.49 | 90.29 | 83.58 | 88.12 |
| CVL+F               | 89.63 | 89.32 | 79.31 | 84.45 |
| Our TCL               | 92.50 | 92.86 | 86.78 | 91.16 |

Min$^1$               | 0.94  | 2.20  | 3.20  | 2.26  

TABLE III: Validation accuracy (%) on LabelMe-AMT and CUBShape datasets. "F" denotes fine-tuning on trusted data.

| Approach                | LabelMe-AMT | CUBShape |
|-------------------------|-------------|----------|
| DL-MV                   | 79.35 (± 0.48) | 91.42 (± 0.18) |
| DL-DS                   | 82.48 (± 0.13) | 92.00 (± 0.16) |
| DL-IBC                 | 81.92 (± 0.07) | 92.07 (± 0.11) |
| DL-CRH                 | 80.12 (± 0.25) | 92.18 (± 0.16) |
| AggNet                  | 84.82 (± 0.16) | 92.18 (± 0.10) |
| CrowdLayer               | 81.42 (± 3.34) | 91.92 (± 0.06) |
| MBEM                   | 79.63 (± 3.74) | 92.06 (± 0.18) |
| Max-MIG                 | 85.60 (± 0.14) | 91.96 (± 0.38) |
| CVL                    | 86.04 (± 0.34) | 91.85 (± 0.10) |
| DL-MV+F                  | 84.21 (± 0.30) | 91.85 (± 0.28) |
| DL-DS+F                  | 86.27 (± 0.13) | 92.21 (± 0.16) |
| DL-IBC+F                | 86.35 (± 0.09) | 92.25 (± 0.22) |
| DL-CRH+F                | 85.32 (± 0.11) | 91.46 (± 0.16) |
| AggNet+F              | 86.04 (± 0.29) | 92.18 (± 0.10) |
| CrowdLayer+F          | 85.38 (± 1.40) | 91.99 (± 0.16) |
| MBEM+F              | 85.55 (± 0.74) | 92.28 (± 0.18) |
| MAX-MIG+F            | 86.19 (± 0.06) | 91.99 (± 0.32) |
| CVL+F               | 86.89 (± 0.25) | 92.06 (± 0.10) |
| Our TCL               | 88.09 (± 0.12) | 92.64 (± 0.06) |

an information theoretic method, which finds the information intersection between two classifiers; and (v) CVL [29] is a coupled-view method, which introduces several effective learning schemes to enhance robustness to label noise. In addition, since our method uses information from clean trusted data, for a fair comparison, we conduct additional fine-tuning on the trusted data based on these pretrained baselines. All the results are reported as the average figures of three trials. In our approach, we use the isotonic regression method [82] or its multi-class version [31] to perform calibration during training.

Tab. II shows the results on four synthetic crowdsourcing CIFAR10 datasets, where some observations can be concluded. First, the common baseline, DL-MV, which directly learns with the aggregated labels by majority voting, performs poorly on all datasets. This implies that effective label aggregation is crucial to improving the performance of the data classifier. Second, we can see that except for DL-MV, the three label aggregation methods are not stable across different datasets, and there is no label aggregation algorithm that outperforms others consistently, which is in accord with the observations in related work [93]. Third, the performance of label aggregation methods without using instance features is weaker than the methods that jointly learn the data classifier and annotator model. Fourth, the simple fine-tuning technique can help, but it is not very effective since the amount of trusted data is small. Last, our TCL approach outperforms other benchmarks on all four datasets under the evaluation of validation accuracy, showing TCL is more data-efficient than employing the fine-tuning technique in the existing methods.

Fig. 5 clearly shows the different iterative learning processes of TCL without and with calibration on the IND-3 dataset. TCL without calibration does not combine prediction into more reliable co-labels, and classifiers eventually fit to more label noise. In contrast, with calibration via trusted data, our TCL approach achieves effective mutual improvement of classifier learning and prediction combination during training, i.e., the pure co-labels make models less overfit to false labels, and the co-labels become purer as the model improves; finally, both reliable co-labels and clean labels are used to retrain a network, which provides more improvement (see Section IV-C). Hence, the great performance of our approach is due to the reliable prediction combination via the effective usage of trusted data.

After the promise is achieved on synthetic datasets, we check the performance on the real crowdsourcing dataset LabelMe-AMT. As reported in column 2 of Tab. III, fine-tuning pretrained baselines on trusted data can improve the best accuracy from 86.04% to 86.89% (increased by 0.85%), while our approach achieves a much better accuracy of 88.09%, with a 2.06% accuracy improvement compared with the best method without trusted data, demonstrating its superiority in the efficient utilization of the small, trusted dataset. Fig. 6 clearly shows the mutual improvement of classifier learning and prediction combination during training on LabelMe-AMT. Therefore, not only is deep classifier learned...
Fig. 4: Comparison between the true confusion matrices and the estimated confusion matrices of three annotators by different methods on the real-world dataset LabelMe-AMT.

Fig. 5: Accuracy of the data classifier, label aggregator and co-labels on our TCL approach without calibration (a) and TCL (b) during iterative training on the IND-3 dataset.

Fig. 6: Accuracy of the data classifier, label aggregator and co-labels on our TCL approach during iterative training on the Labelme-AMT dataset (a) and CUBShape dataset (b).

We further evaluate the performance on another real crowdsourcing dataset, CUBShape (see column 3 of Tab. III). As reported, since the label noise of this dataset is not severe, the DL-MV method also attains a good result, with a validation accuracy of 91.42%, and only a 0.10% improvement (from 92.18% to 92.28%) in the best accuracy is obtained by the fine-tuning strategy. However, compared with other methods, our approach still achieves substantial label noise performance gains (0.46% above 92.18%), which succeeds in performing progressive improvement via trustable co-label updating (see Fig. 6b). Fig. 7 shows the performance advantage of our TCL approach on the above two real-world crowdsourcing datasets more clearly.

B. Learning from Auto-labeling Sources

As mentioned before, the deep learning community has increasing interest in using various automatic labeling sources to collect large-scale labeled datasets, and we can make
The results on four synthetic CIFAR10 datasets under this setting are shown in Tab. IV. Obviously, we can find that our TCLS delivers much better accuracy than other methods on all four noisy datasets. The average minimal improved accuracy reaches 4.97% on four datasets, which empirically demonstrates the advantage of modeling correlated annotators by neural networks. Fig. 8 shows the progressive learning process of TCL+I and TCLS during training on the COR-2 dataset.

The optimizer setting for the data classifier is the same as TCL, and for the label aggregator, we use the Adam optimizer with a batch size of 128, a learning rate of 0.001, and betas of (0.9, 0.999).

For the real SocialEvent dataset, for generality, we use three-layer fully connected neural networks to represent the data classifier, where the first and second hidden layers have 128 and 32 units, respectively (with ReLU activations), and one softmax output layer is on the top. Our TCL approach trains two classifiers for $T = 10$ iterations and retrains the data classifier for 30 epochs. In each iteration round, the data classifier is trained for 5 epochs. We use the Adam optimizer with a batch size of 128, a learning rate of 0.001, and betas (0.9, 0.999). Our TCLS approach uses the same setting as TCL, except that the label aggregator is modeled as a three-layer fully connected neural network, where the first and second hidden layers have 64 and 32 units, respectively, and we use the Adam optimizer with a batch size of 128, a learning rate of 0.001, and betas (0.9, 0.999) for the aggregator.

Results. We compare our approaches TCL and TCLS with the same baselines as the other setting. In addition, as the trusted data in this case include noisy labels, we also validate some baselines with initializing parameters or selecting annotators by such trusted data.
one dataset, but poorly on another dataset; for example, on
four datasets, which means those methods perform well on
not hold, and therefore, their performance is unstable on such
datasets with correlated annotators. "F" denotes fine-tuning on
trusted data. "I" denotes initializing parameters by trusted

denotes fine-tuning on trusted data. "I" denotes initializing
parameters or selecting annotators by trusted data.

TABLE VI: Validation AUC on SocialEvent dataset. "F"
denotes fine-tuning on trusted data. "I" denotes initializing
parameters or selecting annotators by trusted data.

| Approach        | COR-1 | COR-2 | COR-3 | COR-4 |
|-----------------|-------|-------|-------|-------|
| DL-MV           | 83.83 | 58.67 | 84.31 | 85.30 |
| DL-DS           | 84.04 | 46.54 | 82.10 | 86.20 |
| DL-IBCC         | 84.29 | 45.24 | 81.98 | 86.28 |
| DL-CRH          | 84.25 | 46.99 | 82.62 | 84.99 |
| AggNet          | 84.80 | 85.81 | 83.54 | 86.87 |
| CrowdLayer      | 81.41 | 82.39 | 80.43 | 82.70 |
| MBEM            | 84.22 | 85.93 | 83.41 | 85.42 |
| Max-MIG         | 84.27 | 86.40 | 83.53 | 85.58 |
| CVL             | 84.04 | 85.65 | 84.27 | 86.58 |
| DL-MV+I+F       | 83.93 | 77.12 | 84.35 | 85.38 |
| DL-DS+I+F       | 84.04 | 53.24 | 82.10 | 86.20 |
| DL-IBCC+F       | 84.29 | 52.54 | 81.98 | 86.28 |
| DL-CRH+F        | 84.28 | 54.38 | 82.70 | 84.99 |
| AggNet+F        | 84.81 | 86.03 | 83.75 | 87.00 |
| CrowdLayer+F    | 81.46 | 82.59 | 80.63 | 82.73 |
| MBEM+F          | 84.25 | 85.99 | 83.43 | 85.56 |
| Max-MIG+F       | 84.27 | 86.50 | 83.65 | 85.64 |
| CVL+F           | 84.33 | 85.92 | 84.49 | 86.63 |
| AggNet+I+F      | 84.43 | 86.35 | 84.29 | 86.95 |
| CrowdLayer+I+F  | 80.15 | 82.57 | 80.67 | 82.57 |
| Max-MIG+I+F     | 84.31 | 86.22 | 83.40 | 86.22 |
| CVL+I+F         | 84.32 | 86.40 | 84.79 | 86.51 |
| Our TCL         | 84.89 | 87.39 | 83.68 | 87.45 |
| Our TCL+I       | 84.31 | 90.37 | 83.82 | 88.53 |
| Our TCLS        | **90.62** | **93.09** | **93.11** | **91.62** |
| Min↑            | 5.73  | 2.72  | 8.32  | 3.09  |

TABLE VI: Validation AUC on SocialEvent dataset. "F"
denotes fine-tuning on trusted data. "I" denotes initializing
parameters or selecting annotators by trusted data.

| Approach        | Validation AUC(%) |
|-----------------|-------------------|
| DL-MV+I         | 51.25 (± 2.64)    |
| DL-DS+I         | 51.55 (± 0.45)    |
| DL-IBCC+I       | 51.09 (± 1.35)    |
| DL-CRH+I        | 55.07 (± 3.05)    |
| AggNet+I        | 62.55 (± 0.17)    |
| CrowdLayer+I    | 60.44 (± 0.25)    |
| MBEM+I          | 62.91 (± 0.22)    |
| Max-MIG+I       | 56.22 (± 1.45)    |
| CVL+I           | 63.88 (± 0.07)    |
| DL-MV+I+F       | 63.20 (± 1.05)    |
| DL-DS+I+F       | 70.73 (± 1.73)    |
| DL-IBCC+I+F     | 73.79 (± 1.07)    |
| DL-CRH+I+F      | 72.57 (± 1.79)    |
| AggNet+I+F      | 66.70 (± 3.75)    |
| CrowdLayer+I+F  | 72.81 (± 1.58)    |
| MBEM+I+F        | 73.18 (± 0.78)    |
| MAX-MIG+I+F     | 61.73 (± 0.88)    |
| CVL+I+F         | 73.00 (± 0.97)    |
| Our TCL         | 69.14 (± 6.38)    |
| Our TCL+I       | 74.05 (± 0.69)    |
| Our TCLS        | **75.21 (± 0.24)** |

In addition, we can obtain some other observations from
these results. First, the correlated annotators make the labeling
assumption of other state-of-the-art methods (including TCL)
not hold, and therefore, their performance is unstable on such
four datasets, which means those methods perform well on
one dataset, but poorly on another dataset; for example, on

COR-3 dataset, most approaches are inferior to the simple DL-
MV approach with the fine-tuning, which achieves a validation
accuracy of 84.35%. Second, similar to the performance on
generated crowdsourcing datasets, a simple fine-tuning strat-

gy has a limited effect on improving the good pretrained
baselines. Third, initializing confusion matrices or pseudo
labels via trusted data for baselines does not always work well
on these four datasets, which may be because all annotators
are positive and initialization by majority voting is enough at
most times.

We also evaluate the performance on the real dataset So-
SocialEvent with multiple auto-labeling sources. The results are
shown in Tab. VI. The SocialEvent dataset has many adver-
saries who deliberately give many incorrect labels, and their
labeling accuracies are even lower than the prior probability
of the negative class (85%). Therefore, those methods without
any trusted data all fail, and we only present the results
with initializing parameters or selecting annotators by trusted
data. Faced with such heavy label noise, a simple fine-tuning
technique also achieves great performance improvement.
In addition, between those methods using trusted data, TCLS
achieves better performance, with an AUC of 75.21%, again
verifying its advantages compared with the other state-of-
the-art methods. Fig. 9a shows the validation AUC of two
classifiers in TCL+I and TCLS after alternate optimization,
from which we can verify that the performance of the label
aggregator benefits from the high modeling capacity of deep
networks during iteration, also making the data classifier
learning more effective.

C. Ablation Study

We mainly study the impact of different components on
TCL, and the results on TCLS are similar.

Impact of calibration. To study the impact of calibration,
we use the four generated CIFAR10 datasets with indepen-
dent annotators to perform the evaluation. Fig. 9b shows the
average accuracy of our TCL approach with and without

calibration (TCLnoCa) after the alternate optimization stage
on four datasets, which verifies the claim that calibration
via trusted data plays a critical role in guiding the trustable
prediction combination and that our approach utilizes trusted
data effectively.

Size of trusted data. Trusted data provide important extra
information for TCL to guide the learning process. To study
the impact of the size of trusted data, we validate the per-
formance of TCL under different sizes of trusted data on the
IND-3 dataset. To decouple the effect of calibration and clean
sample training, we also validate the performance of TCL with
retraining only on noisy data (TCLReTN). As shown in Fig.
9c, although our approach is more data-efficient than the fine-
tuning technique, the performance of our approach will decline
if the size of trusted data is too small.

Impact of retraining strategy. To study the impact of the
retraining strategy on generated crowdsourcing datasets IND-
3 and IND-4, we compare the performance of TCL, TCL with
retraining on noisy data with co-labels (TCLReTN), and TCL

8}
In this paper, we propose a **Trustable Co-label Learning** approach to make full use of a small amount of trusted data to facilitate robust classifier learning from multiple annotators. It alternately refines classifiers and relabels pseudo labels with the guidance of a small trusted dataset based on the conditionally independent assumption. In addition, we further improve this approach by modeling the label aggregator as a multilayer neural network to enhance modeling ability when all instances are labeled by every annotator. The proposed approach achieves superior performance in terms of effectiveness and robustness on both synthetic and real datasets. In the future, we will explore a more complicated but still trainable probabilistic aggregator modeling way to represent the dependence among different annotators when there are missing labels.

**Acknowledgements.** This work was partially supported by grants from the National Key Research and Development Plan (2020AAA0140001), National Natural Science Foundation of China (61772513), and Beijing Natural Science Foundation (19L2040). Shiming Ge is also supported by the Youth Innovation Promotion Association, Chinese Academy of Sciences.

**Impact of co-label initialization on TCLS.** Our TCLS approach uses trusted data to train a Naive Bayes classifier to initialize co-labels. To study this effect, we conduct majority voting to initialize co-labels and perform experiments on COR-2 and SocialEvent datasets. After alternate optimization, TCLS achieves an accuracy of 93.04% on the COR-2 dataset, and TCLS with co-label initialization by majority voting achieves an accuracy of 92.87%. TCLS attains an AUC of 66.13% on the SocialEvent dataset, while TCLS with co-label initialization by majority voting attains an AUC of 63.41%. These results are consistent with other baselines, which shows that having good initialization is important when there are many adversaries in annotators, and initialization by majority voting is not undesirable in the usual case.

**V. CONCLUSION**

In this paper, we propose a **Trustable Co-label Learning** approach to make full use of a small amount of trusted data and trusted data make positive contributions to the retraining stage.

**REFERENCES**

[1] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[2] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Neural Information Processing Systems (NeurIPS)*, 2015, pp. 91–99.

[3] H. Chen, F. X. Han, D. Niu, D. Liu, K. Lai, C. Wu, and Y. Xu, “MIX: multi-channel information crossing for text matching,” in *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2018, pp. 110–119.

[4] Z. Sun, Q. Zhang, Y. Li, and Y. Tan, “Dppdl: A dynamic parallel-partial data layout for green video surveillance storage,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 1, pp. 193–205, 2018.

[5] G. Wang, X. Liu, C. Li, Z. Xu, J. Ruan, H. Zhu, T. Meng, K. Li, N. Huang, and S. Zhang, “A noise-robust framework for automatic segmentation of COVID-19 pneumonia lesions from CT images,” *IEEE Transactions on Medical Imaging*, vol. 39, no. 8, pp. 2653–2663, 2020.

[6] Y. Yao, J. Zhang, F. Shen, X. Hua, J. Xu, and Z. Tang, “Exploiting web images for dataset construction: A domain robust approach,” *IEEE Transactions on Multimedia*, vol. 19, no. 8, pp. 1771–1784, 2017.

[7] H. Wang, X. Wu, and Y. Jia, “Video annotation via image groups from the web,” *IEEE Transactions on Multimedia*, vol. 16, no. 5, pp. 1282–1291, 2014.

[8] C. Chaudhary, P. Goyal, D. N. Prasad, and Y. P. Chen, “Enhancing the quality of image tagging using a visio-textual knowledge base,” *IEEE Transactions on Multimedia*, vol. 22, no. 4, pp. 897–911, 2020.

[9] Y. Lin and H. Chen, “Tag propagation and cost-sensitive learning for music auto-tagging,” *IEEE Transactions on Multimedia*, vol. 23, pp. 1605–1616, 2021.

[10] W. Li, L. Wang, W. Li, E. Agustsson, and L. V. Gool, “Webvision database: Visual learning and understanding from web data,” *CoRR abs/1708.02662*, 2017.

[11] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, “Understanding deep learning requires rethinking generalization,” in *International Conference on Learning Representations (ICLR)*, 2017.

[12] T. Liu and D. Tao, “Classification with noisy labels by importance weighting,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 10, pp. 3307–3309, 2016.

[13] J. Cheng, T. Liu, K. Ramamohanarao, and D. Tao, “Learning with bounded instance-and label-dependent label noise,” in *International Conference on Machine Learning (ICML)*, 2020, pp. 1789–1799.

[14] 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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 6835–6846.

[15] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, “ImageNet: A large-scale hierarchical image database,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 248–255.

[16] J. Biel and D. Gatica-Perez, “The youtube lens: Crowdsourced personality impressions and audiovisual analysis of vlogs,” *IEEE Transactions on Multimedia*, vol. 15, no. 1, pp. 41–55, 2013.
