Robust (Deep) learning framework against dirty labels and beyond

Ghiassi, Amirmasoud; Younesian, Taraneh; Zhao, Zhilong; Birke, Robert; Schiavoni, Valerio; Chen, Lydia Y.

DOI
10.1109/TPS-ISA48467.2019.00038

Publication date
2019

Document Version
Final published version

Published in
Proceedings - 1st IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications, TPS-ISA 2019

Robust (Deep) Learning Framework Against Dirty Labels and Beyond

Amirmasoud Ghiassi*, Taraneh Younesian*, Zhilong Zhao†, Robert Birke‡, Valerio Schiavoni§ and Lydia Y. Chen*
†University of Grenoble, France. E-mail: zilong.zhao@grenoble-inp.fr
‡ABB Future Labs, Switzerland. E-mail: robert.birke@ch.abb.com
§University of Neuchâtel, Switzerland. E-mail: valerio.schiavoni@unine.ch
*TU Delft, The Netherlands. E-mail: s.ghiassi@tudelft.nl,t.younesian@tudelft.nl,lydiaychen@ieee.org

***Abstract*—Data is generated with unprecedented speed, due to the flourishing of social media and open platforms. However, due to the lack of scrutinizing, both clean and dirty data are widely spreaded. For instance, there is a significant portion of images tagged with corrupted dirty class labels. Such dirty data sets are not only detrimental to the learning outcomes, *e.g.*, misclassified images into the wrong classes, but also costly. It is pointed out that bad data can cost the U.S. up to a daunting 3 trillion dollars per year. In this paper, we address the following question: how prevailing (deep) machine learning models can be robustly trained given a non-negligible presence of corrupted labeled data. Dirty labels significantly increase the complexity of existing learning problems, as the ground truth of label's quality are not easily assessed. Here, we advocate to rigorously incorporate human experts into one learning framework where both artificial and human intelligence collaborate. To such an end, we combine three strategies to enhance the robustness for deep and regular machine learning algorithms, namely, (i) data filtering through additional quality model, (ii) data selection via actively learning from expert, and (iii) imitating expert's correction process. We demonstrate three strategies sequentially with examples and apply them on widely used benchmarks, such as CIFAR10 and CIFAR100. Our initial results show the effectiveness of the proposed strategies in combating dirty labels, *e.g.*, the resulting classification can be up to 50% higher than the state-of-the-art AI-only solutions. Finally, we extend the discussion of robust learning from the trusted data to the trusted execution environment.**

***Index Terms*—Deep neural networks, dirty labels, data filtering, active learning, trusted execution, adversarial learning**

I. Introduction

The ever-increasing self-generated contents on social media, e.g., Instagram images, power up the deep neural networks, but also aggravate the challenge of noisy and corrupted data. Large portion of images accessible on the public domain come with labels which are unfortunately dirty due to careless annotations [1], [2], cheap data curation or even adversarial strategies [3]–[5]. Consider Google Image Search to curate training data set [6]. The image search can conveniently return a large number of images whose auxiliary information, e.g., semantic text, contains the searched terms. With non-negligible probability, a large number of unrelated images could be thus included, especially for less popular queries.

Deep neural network models have advanced greatly in recent years due to ever increasing computational capacity, e.g., GPU, algorithmic breakthroughs, e.g., generative adversarial networks [6], in addition to the fuel of big data. Deep models, such as convolutionary network, can solve complex image classification tasks and reach remarkable accuracy, compared to the standard machine learning models. Unfortunately, there are several studies pointing out the weakness of deep models against dirty data, from the corrupted images to the labels [7]–[9]. The high learning capacity of deep networks can memorize the images structures of clean and also corrupted label labels [10] due to the memorization effect of networks. Classification accuracy on standard image benchmarks unfortunately degrades drastically in the presence of dirty labels. For example [10], the accuracy of using trained AlexNet to classify CIFAR10 images drops from 77% to 10% with random labels.

Indeed, dirty label is a long-standing challenge for statistical methods whose model training process depends on the labels. The limited availability of ground truth about the label quality significantly hardens the learning problems. As a result, the central theme of a large body of the related work is to distill the influence of dirty labels in the model training process without the ground truth knowledge of labels - unsupervisedly learning the label qualities. To avoid being polluted by dirty labels in the model training phase, auxiliary statistical filtering mechanisms [11], [12] are applied to remove or replace the suspicious data for the original model training. In other words, multiple different learning models are applied and the classification outcomes are determined by their joint consensus. Specifically, the recent trend to enhance robustness of deep networks by designing novel network architecture, e.g., [12] has two parallel networks cross training each other, or by modifying the loss function that is aware of the presence of dirty lables [13]. However, the promising results of robust deep networks in countering the dirty labels are at the cost much higher computational overhead.

Our vision to address the dirty data issue is through the collaboration with human experts, designing a learning framework where artificial intelligence and human experts co-teach each other. Such learning framework consists of three key components: (i) quality model, (ii) active learning with human experts, and (iii) CopyNet, a system which imitates experts. The quality model [14] is an additional classifier which filters out the suspicious data that might have the corrupted labels

978-1-7281-6741-1/19/$31.00 ©2019 IEEE
DOI 10.1109/TPS-ISA48467.2019.00038

without the ground truth of the label quality. The choice of such classifier depends on the data sets as well as the actual learning tasks. Those suspicious data can be cleansed through the predictions of the quality model or human expert which can provide the ground truth. The cleansing cost of human experts is much higher than the model-based solution. The research question here is to define and select suspicious data, given a certain budget to acquire data, being from cheap crowd source or expensive ground truth data provided by human experts.

Ultimately, the proposed learning framework imitates the correction process of human experts and formulates a rigorous learning problem, termed CopyNet. Different from the main stream of robust deep networks against the dirty labels, CopyNet employs Amateur (a convolution neural network) to first classify images based on the given (partially corrupted) labels and Expert (another neural network) to correct the labels based on the expert's inputs. In other words, CopyNet uses a small fraction of ground truth provided by human experts in the training phase and leverages the given labels as auxiliary information in the inference phase. Our preliminary results show that such a collaboration learning framework of humane and artificial intelligence can outperform the state of the art robust deep networks which only relies on techniques of artificial intelligence. Finally, we conclude this study by extending the discussion from the trusted data to the trusted execution environment for robust machine learning frameworks.

II. Quality Models

A. System Model

We focus on the online-learning scenario, in which data instances continuously arrive at the learning system over time in batches. All the data instances arriving at system are labelled. Labels can be correct (i.e. clean label) or incorrect (i.e. noisy label). $\mathcal{D}_i$ denotes the data batch comes to system at time t_i, it has labels Y_i. To kick-start the learning process, we assume batch $\mathcal{D}_0$ has only clean labels data. We also assume that we have a small data set P for which we know its true labels, and that we use to test the accuracy of our system at the end of learning process of each epoch.

B. Framework

Inspired by RAD (Robust Anomaly Detection) framework [14], we change RAD's training strategy by only using the data selected from the current batch to update models. We also let classifier join the process to filter out cleansed data, that makes a double selection system. The RAD Duo framework is depicted in Fig. 1. The system comprises two components. A label quality model $\mathfrak{L}$ (label quality predictor) aims at discerning clean labels from dirty labels and a classifier model $\mathfrak{C}$ (anomaly classification) targets the specific classification task at hand.

Quality model is used to determine if the data instances are correctly or incorrectly labelled, and to prevent the classifier to be over-fitting to noise. $\mathfrak{L}_{i-1}$ is the quality model which is trained with previous data batches up to time t_{i-1}. When $\mathcal{D}_i$ comes, $\mathfrak{L}_{i-1}$ will do the prediction on it. Comparing the prediction and the given labels Y_i, we could divide $\mathcal{D}_i$ into two parts: (1) $\mathcal{D}_i^*$: the data instances in $\mathcal{D}_i$ whose prediction and given label are identical, *i.e.*, a trusted dataset; (2) $\mathcal{U}_i$: data whose prediction and given label are different, *i.e.*, a suspicious dataset. Quality model will keep $\mathcal{D}_i^*$ locally, and send $\mathcal{D}_i^*$ and $\mathcal{U}_i$ to classifier.

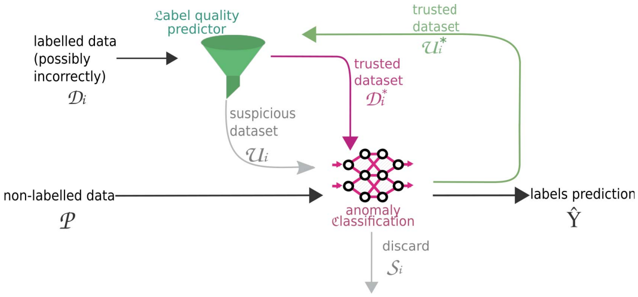


Fig. 1: RAD Duo framework.

$\mathfrak{C}_{\mathrm{i}-1}$ is the classifier trained until time t_{i-1}. Once it receives $\mathcal{D}_i^*$ and $\mathcal{U}_i$ from $\mathfrak{L}$, it keeps $\mathcal{D}_i^*$ locally, and do the prediction on $\mathcal{U}_i$, then $\mathcal{U}_i$ will also be divided into two parts: (1) $\mathcal{U}^*$: the data instances in $\mathcal{U}_i$ whose predictions are same as their given labels, this part of data is also called trusted dataset; (2) $\mathcal{S}_i$: the data instances in $\mathcal{U}_i$ whose predicted labels are different from their given labels. The classifier keeps the $\mathcal{U}_i^*$ locally and also send it back to quality model. $\mathcal{S}_i$ will be discarded.

Now, there are data $\mathcal{D}_i^*$ and $\mathcal{U}_i^*$ in both quality model and classifier, they will use these data to update their models locally to get $\mathfrak{L}_\mathrm{i}$ and $\mathfrak{C}_\mathrm{i}$, then they could release $\mathcal{D}_i^*$ and $\mathcal{U}_i^*$ from their buffers.

Practically, inference time of quality model should less than classifier, because when a data batch $\mathcal{D}_i$ comes, quality model will make the predictions on the whole batch, and classifier will only do the predictions on a small fraction of $\mathcal{D}_i$. In general, we want to keep the data selection time as low as possible.

C. Use-cases and Datasets

In order to demonstrate the broad applicability of RAD Duo, we consider the following three use-cases: (i) CIFAR-10 [15], (ii) CIFAR-100 [16] and (iii) FaceScrub [17].

CIFAR-10 and CIFAR-100 are two wildly used datasets for image classification. CIFAR-10 contains 60'000 images evenly distributed in 10 classes. There are 50'000 training images and 10'000 test images. The test dataset contains exactly 1000 randomly-selected images from each class. CIFAR-100 is similar as CIFAR-10, except its 60,000 images are evenly distributed in 100 classes. In test dataset of CIFAR-100, each class contains 100 images, and the remaining images are training dataset.

The FaceScrub dataset is used for face recognition. It originally contains more than 100'000 face images of 530 celebrities, with about 200 images per person. Male and female images are equally represented. After we manually check the dataset to filter out some repeated and unclear images, we extract a representative subset of 3.3k images. This subset covers 20 people with the highest number of images, *e.g.* 12 males and 8 females. As the original FaceScrub face images are retrieved from internet, the image sizes are different, we re-scale all images to the same 128×128 pixel format. The only label we use afterwards is the name of the person.

D. Experimental Setup

The RAD Duo algorithm is implemented using Keras [18]. The setting of three datasets for experiments are summarized in Table I.

We evaluate RAD Duo against CIFAR-100 [15], CIFAR-10 [16] and FaceScrub [17]. All the code is implemented leveraging Keras.

Noise. We inject the noise into the three datasets by switching the true label to at random amongst the other ones. The noise is symmetric, *i.e.*, when we change a label, the probabilities to any other labels are equal. We inject noise only on training data, no noise on test data.

Quality and Classification model. For CIFAR-10 and FaceScrub dataset, we use a ResNet [19] model for quality model and a VGG [20]-like CNN (Convolutional Neural Network) for classifier. Even though quality model has more layers than classifier, it uses fewer total parameters. For CIFAR-100, as this dataset has more classes, we use ResNet as quality model and a deeper ResNet as classifier. While we considered other structures and CNN models, these combinations give best balance between final accuracy and training time.

Baselines. The proposed RAD Duo is compared against two baseline data selection schemes: (i) *No-Sel*, where all data instances of arriving batches are used for training the classification model; and, (ii) *Full-Clean* which emulates an omniscient agent who can perfectly distinguish between clean and noisy labels, and could recover all the noisy labels. The two baselines are representative of the worst and best possible data selection strategies. We expect RAD Duo to fall in between these two extreme cases.

E. Evaluation of Result

Fig. 2a shows three curves extracted from our experiments on CIFAR-10. From the No-Sel curve, we observe that is noisy labels are indeed destructive to classifier. Secondly we could see that under 30% noise, RAD Duo still follows the trend of Full-Clean, despite strong deviations, at the end of each batch training epochs, it could always recover to a higher or similar level than the previous batch. We can observe the similar situation in [21] with different on-line learning setting. The reason that all the curves suffer a periodic up-down pattern, is because our model is optimized on the current training data: different batches provide different *subviews* of the data, and the empirical distribution can be different. When a new data batch comes, we will generate a gradient based on new data, but applied on the remaining model, that could influence the accuracy. Furthermore, we reset the time-decayed learning rate when a new batch comes. Therefore, even if all the batches follow the same distribution, the system could temporarily wander off from the previous optimum.

TABLE I: Dataset description

Use case	CIFAR-100	CIFAR-10	FaceScrub
#trainig data instances	50,000	50,000	2,639
#test data instances	10,000	10,000	665
#classes N	100	10	20
#features f	32*32	32*32	128*128
data batch size	10, 000	1, 000	200
initial batch $\mathcal{D}_0$ size	10,000	10,000	639
training epochs for $\mathcal{D}_0$	60	60	60
training epochs except $\mathcal{D}_0$	60	10	20

TABLE II: Evaluation of the RAD Duo on different datasets under 30% noise

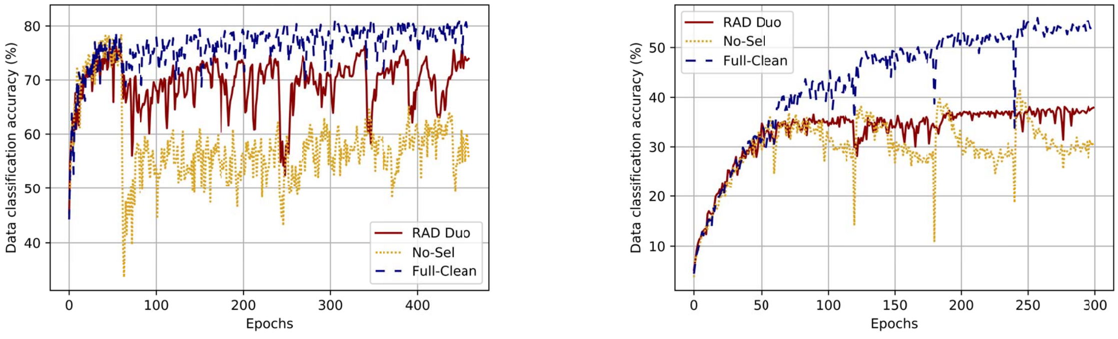
Dataset	No-Sel	Full-Clean	RAD Duo	Improvement
Cifar-100	30.01%	53.61%	33.99%	3.98%
Cifar-10	55.63%	81.26%	73.89%	18.26%
FaceScrub	43.16%	67.07%	46.47%	3.31%

* All the results are averaged on 3 times experiments

Fig. 2b presents our results on CIFAR-100. The Full-Clean and RAD Duo curves are continuously increasing even though with the presences of oscillations. We observe how in the No-Sel case, the arriving of a new batch leads to a drop on accuracy, followed by an increase to a higher level, even better than RAD Duo. This is because comparing to RAD Duo, No-Sel could use more data to train. As the training process continues, its accuracy will eventually decrease, this is because the model begins to be over-fitting on the noisy training data.

Final accuracy results for all experiments are summarized in Table II, the column of improvement is from the comparison between RAD Duo and No-Sel. It shows that for CIFAR-10 dataset, the improvement is significant. While for CIFAR-100 and FaceScrub, the improvements are weaker. From the column Full-Clean, we can see that even without noisy label data, the final accuracy of CIFAR-100 and FaceScrub are not that good, that is because CIFAR-100 has more classes, and the image size of FaceScrub are 16× than CIFAR-100 and CIFAR-10, that makes them take longer time to converge. As the whole system's accuracy is low, two models of RAD Duo will let too many noisy data pass or they will discard too many training data, the selection could not work with the models with that low accuracy. These two cases are all bad for RAD Duo.

In summary, for CIFAR-100 and FaceScrub, our system has still large margins of improvements. Toward the goal of improving the structure so that even if accuracy is low, our model could converge closer to Full-Clean within the same training epochs. We describe these improvements in §II-F.



(a) With CIFAR-10 dataset

(b) With CIFAR-100 dataset

Fig. 2: Evolution of learning accuracy over time under 30% noise

F. Future work beyond RAD Duo Framework

Though RAD Duo improves final accuracy for datasets, we report on the current limitations: (1) for quality model, it is very useful to filter out uncertain data, but training an extra CNN except of classifier is also expensive. If the computing resources are limited, it may not be possible to train two CNN between the interval of two on-line learning batches, typically in the order of second/millisecond time unit; (2) according to Fig. 2a, we could see that the classifier's accuracy improves over time (even though there are oscillations), if the computing resources and time interval between batches are sufficient, we can buffer the $\mathcal{S}$, we know that there are some mis-discarded data inside $\mathcal{S}$ due to the limited performance of former models. We could set a limit size of buffer of accumulated $\mathcal{S}$, once the buffered $\mathcal{S}$ are over the limit, we attach these data to $\mathcal{D}$ of next batch, re-do the selection process on buffered $\mathcal{S}$ together with $\mathcal{D}$ and release local buffer.

III. Active learning from Experts

Discarding data instances that the quality model filters out as noisy could result in a too high loss of information. Instead, the idea is to try to make use of the noisy data instances by cleansing them with the help of a human expert, *i.e.*, an oracle. However asking the oracle for the true labels of the whole is too expensive and time consuming. Therefore, we leverage active learning to identify the important and informative noisy samples which have the highest impact on the performance of the classifier if their true labels are available. Here we focus on designing such a systems in the same online batch-arrival setting as described in §II.

A. Framework

Similar to the RAD Duo framework described in Section II, our model, termed QAL , consists of a label quality model and a classifier model plus an active learner component. The active learner decides which noisy samples to send to the oracle for cleansing. Fig. 3 presents the architecture of QAL .

Similar to RAD Duo learning, $\mathcal{D}_i^*$ is the subset of the current batch that has the predicted label by the quality model equal to the given label, and $\mathcal{U}_i$ is the subset of suspicious data where the two labels differ. The role of the active learner is to choose from $\mathcal{U}_i$ the samples for which the quality model is the least certain on the predicted label and send these to the oracle to be relabelled with their true label.

As quality model we use a multi-class SVM with the decision function $f : \mathbf{x} \to N \times C$ in a one-vs-rest model [22], where N is the number of samples in the batch and C is the number of classes. The decision function represents the distance of each sample from the classification decision boundary. Therefore, we define our measure of uncertainty based on this distance: the shorter the distance to any of the C decision boundaries the higher the uncertainty. We choose the most uncertain instance among $\mathcal{U}_i$ that has the lowest decision function.

B. Experimental Setup

Datasets. We evaluate QAL on the following datasets: *pendigits* (16 features, 10 classes), *usps* (256 features, 10 classes) and *optdigits* (64 features, 10 classes). The datasets are from UCI repository [23] used for handwritten digit recognition.

For all datasets we start with an initial clean set of 150 instances. To speed up training, we limit the data size to $N = 1050$ samples (including the initial set).

Noise. We use the same symmetric noise model as in §II-D but with the following two noise ratios: 60% and 80%. The noise is injected to the labels during training but the test set is assumed to be clean.

Model Details. We use a SVM for both the quality and classifier models since it is a well known model that has been studied with active learning. Our prototype is implemented in Python using the multi-class SVM in *scikit-learn* [24]. We query the true label of the 5 most uncertain noisy samples per batch via the oracle. We repeat each setting 100 times and report the average test accuracy for the classifier.

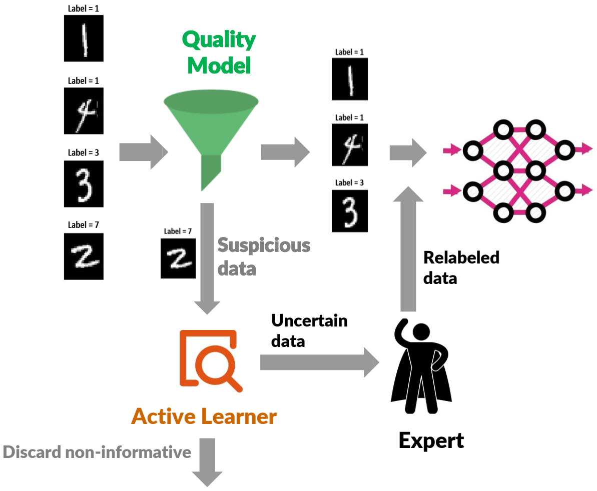


Fig. 3: QAL Architecture

Baselines. To better show the effectiveness of our proposed QAL method we compare it with four other data selection baselines:

- ***No-Sel***, same as in RAD Duo, using all sample to train the classifier with no selection.
- ***Q-only***: without the active learner to query for relabelling suspicious samples, i.e., only the clean data identified by the label comparator are used to train the classifier.
- ***AL-only***: without the quality model where all data is used to training the classifier except some informative instances that are relabel by the oracle.
- ***Opt-Sel***: which assumes a perfect quality model able to distinguish between clean and noisy labels and uses all the clean samples for training.

For a fair comparison, in *AL-only* we also use five active queries per batch.

C. Evaluation

Figure 4 and 5 shows the results for the different selection methods and their comparison with our proposed QAL with uncertainty sampling. As the results show, our model outperforms all the baselines except *Opt-Sel* where there is only clean samples for training. As the noise increases, *No-Sel* causes degradation in the performance of the system, although in lower noise rate the performance of this method is the lowest among all. Moreover, as shown in the figures, in most cases except *pendigits AL-only* has the next lowest performance which shows the effectiveness of the quality model in filtering the noisy samples. This method achieves better results in lower noise rate, *i.e.*, 60% due to the absence of quality filtering. *Q-only* has a better performance than *AL-only*, however, compared to our proposed QAL does not have much improvement from the initial batch specially in the 80% noise. Furthermore, in the higher noise rate does not affect the performance of our method. Although the quality model is not strong enough to separate clean from noisy samples, relabelling the informative suspicious instances using active learning overcomes the noise effect.

To conclude, our results on several datasets show that by leveraging the information in the noisy samples with the help of a human expert we can improve the robustness of the classification model to label noise and reach near the performance of optimal selection where there is only clean samples to train the classifier. This gain in accuracy is achieved only by relabelling 10% of the data instances in each batch.

D. Future work beyond QAL

In this paper we focused on one measure of uncertainty which chooses the data instances closer to the decision boundary. However, this measure disregards the relation between the classes and only focuses on the most uncertain class. Margin sampling is another method for active query selection which selects the data instance that its classification score between two classes are very close to each other, i.e. the model is not certain which of the two best classes to predict. Our preliminary results on this method show significant improvement over the most uncertain method. Due to the space limit, we leave the exploration of this uncertainty measurement for our future work.

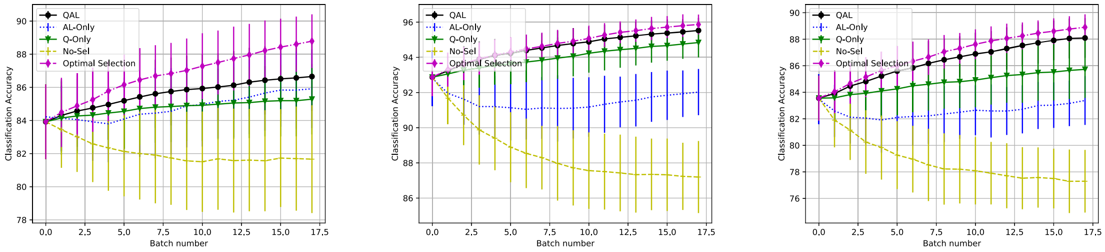
Querying the oracle is expensive, and typically the available budget to relabel queries is limited. One of the biggest challenges in online settings is to decide how much budget to spend at each data batch. Most studies on active learning in the online setting focus on a fixed number of active queries per batch, however they fail to consider the dynamic of the system's performance caused by noise. The accuracy of the quality model and the classification model can change in each batch due to the existence of noise in the labels. Therefore, a dynamic rate for query selection in each batch which considers the total budget can be more effective to improve the classification accuracy. We plan to further explore dynamic active learning and methods to define the dynamic query rate in our future work.

IV. Imitating Experts

According to active learning, the expert provides knowledge about ground truth. Therefore each sample has an expert-evaluated correct label in addition to its noisy label. The presence of both a clean and noisy label for each data sample leads to find the relation between them. In this model, we use noisy labels as extra features to achieve a robust classifier. Also, our network reduces the noise ratio in the dataset by deriving the relation between noisy and clean data.

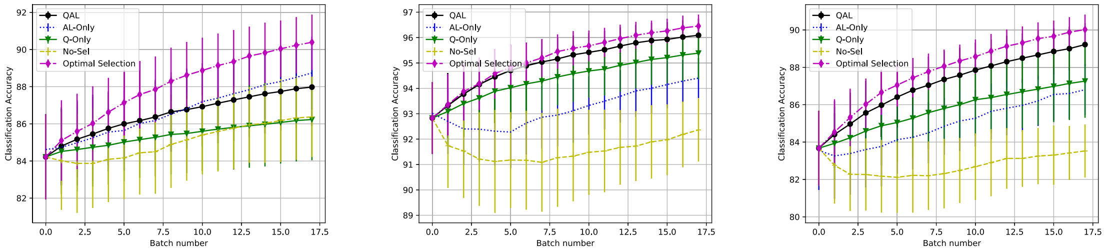
A. Framework

In contrast to prior learning methods, especially image classification, the main idea implemented by CopyNet is to employ dirty labels as part of the training beside ground truth. As shown in Fig. 6, both the ground truth joins and noisy labels are used as auxiliary input to directly learn the corrupted



(a) pendigits (b) optdigits (c) usps

Fig. 4: Results for 80% noise, comparison between the proposed QAL and the baselines.



(a) pendigits (b) optdigits (c) usps

Fig. 5: Results for 60% noise, comparison between the proposed QAL and the baselines.

label dynamics. Towards that end, the ground truth of labels is provided by human experts as a part of the training input. The neural networks are trained using ground truth and noisy labels at the same time. Essentially, for the testing process, images and limited noisy labels act together as input to classify the images.

At its heart CopyNet comprises two cascaded neural networks with feedback, as highlighted in Fig. 6. The first neural network acts as image classifier. It is trained using the images as input and predicted labels from the second neural network. The task of this second neural network is instead to find the relation between noisy labels and ground truth and, hence, correct the prediction of the image classifier. Via the feedback it aims to avoid, during training, the detrimental effects of the noisy labels on the performance of the image classifier. From this point of view, we consider the second neural network as a helper for the first network to improve the test accuracy and recover the true labels from the noisy dataset. This is achieved by learning the pattern of noise via some ground truth during training.

B. Experiments Setup

Datasets. We conduct experiments on MNIST [25], as well as the previously presented CIFAR-10 and CIFAR-100 datasets. MNIST comprises 60’000 examples plus 10’000 samples for testing of handwritten digits to be classified into 10 classes.

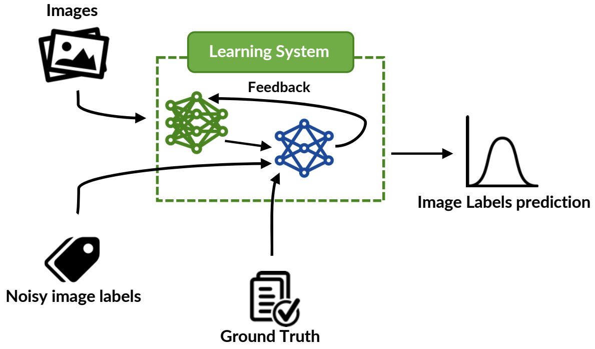


Fig. 6: Training process with noisy labels and ground truth

Noise. We use the same symmetric noise model as described in §II-D with 20%, 30% and 40% noise ratios.

Model Details. We use a 12-layer CNN architecture with ReLU activation functions as image classifier. The second neural network is instead a simpler 4-layer feed-forward network. All networks are implemented using Keras v2.2.4 and Tensorflow v1.13.

Baselines. We compare CopyNet against four baselines from related work, which aim to find the influence of noisy labels on test accuracy.

D2L [13]: modifies the loss function based on the Local Intrinsic Dimensionality (LID) score which detects points of noisy data.

Co-teaching [12]: exchanges the more useful samples between two neural networks. The instances with small-loss are used to train networks. Therefore neural networks teach each other.

Bootstrap [7]: predicts image labels based on a weighted combination of the original label and the model prediction.

Forward [26]: trains model with the labels which are derived from a transition matrix between noisy and clean labels.

C. Evaluation

Fig. 7 presents the classification accuracy on the CIFAR-10 dataset for different noise ratios and amounts of training data: all and halved.

Using all training data CopyNet achieves 89%, 86% and 83% test accuracy for 20%, 30% and 40% of noise ratio, respectively (see Fig. 7a). The accuracy decreases smoothly when increasing the noise ratio. However, higher noise ratio also decrease the convergence speed postponing the time at which the model reaches the steady state accuracy. According to the result, 60 epochs are sufficient to reach a stable accuracy with 20% noise ratio, whereas 170 are needed for 40% noise ratio.

Training with half the data is more difficult which results in a loss in accuracy. Fig. 7b shows that the model achieves accuracy 86%, 82% and 77% accuracy for 20%, 30% and 40% of noise ratio, respectively. However the convergence speed is faster. Here CopyNet is able to reach the steady state in 50 and 130 epochs for 20% and 40% of noise ratio, respectively. This is a 16.7% and 23.5% decrease in convergence time.

Table III compares the accuracy of CopyNet against our baselines and datasets under 30% noise ratio. Among all three datasets, MNIST is the easiest due to the low image complexity. Here our model achieved exceptional test accuracy, even with half of training data. For CIFAR-10 CopyNet achieves 88.30% and 83.15% with 100% and 50% training data, respectively, which is up to 31.83% better then the competing baselines. Similarly, for CIFAR-100, the most complex classification problem, our achieved accuracy are 79.92% and 74.35% with 100% and 50% training data, respectively. This is up to 42.08% better than the rest. Overall we observe that our model is both more data efficient and better at handling more complex classification problems.

D. Future work beyond CopyNet

The core idea of CopyNet is to leverage a fraction of the ground truth of noisy labels and imitate how experts correct such noise data. However, in the real world scenarios, noisy labels exhibit dynamic patterns, i.e., the noisy ratios fluctuate. This presents a new challenge to CopyNet that how to select representative noisy data in both training and testing phase, reflecting truthfully the reality. In other words, it becomes critical to find the general representation of CopyNet for a wide range of noise scenarios encountered in real life learning problems.

TABLE III: Evaluation of CopyNet on different datasets with 30% noise

Dataset	Training Data	Our model	D2L	Co_teaching	Bootstrap	Forward
MNIST	100%	99.38%	86.15%	95.72%	79.47%	95.33%
	50%	99.07%	80.87%	95.36%	51.45%	57.21%
CIFAR-10	100%	88.30%	82.45%	80.29%	77.14%	81.68%
	50%	83.15%	77.06%	76.76%	51.32%	58.39%
CIFAR-100	100%	79.92%	51.13%	45.68%	44.99%	54.18%
	50%	74.35%	42.11%	35.68%	32.27%	48.34%

To such an end, we will resort to the techniques of transfer learning, which aims to find the general structure within different noise patterns. Specifically, we plan train the CopyNet on a large data set of one specific noise patterns and then generalize it to different patterns through retraining weights. We believe that the transferred learning techniques can facilitate CopyNet to accommodate to dynamic noise patterns with an advantage of computational efficiency. .

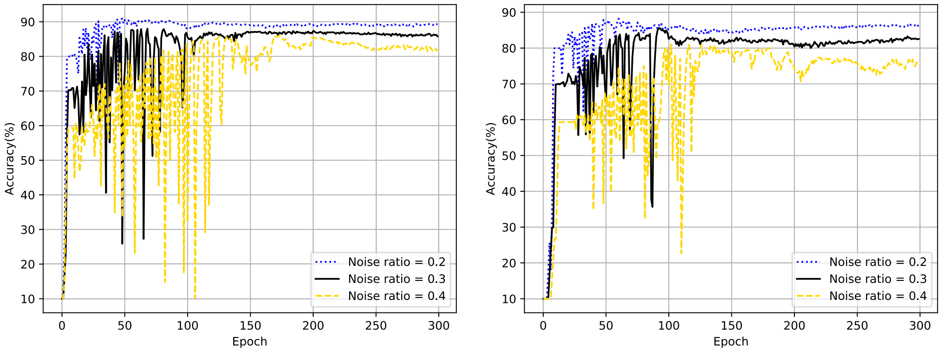
V. Trusted Execution

When considering the robustness properties of ML systems, one aspect often overseen is the actual execution environment in which the training and inference occur. While cloud computing becomes the standard platform *de-facto*, with major IT providers offering built-in support for popular ML frameworks (*e.g.*, Azure Machine Learning [27], Google AI Platform [28], or AWS Machine Learning [29]), in this section, we take the stance that cloud computing can be fundamentally broken from a trustworthiness standpoint. In fact, classical cloud computing environments are ideal playground for both external or internal attackers trying to ex-filtrate the sensitive models being built.

It is well known how pure-software solutions (*e.g.*, fully or partially homomorphic encryption schemes) are far from being practical, and still achieves results in the orders of magnitude weaker than non-encrypted results. While research prototypes exist for confidential machine-learning systems [30], [31], the practical applicability of such schemes remain to be proven over large datasets such as those described earlier.

The recent introduction of trusted execution environments (*TEE*) into mass-market processors, such as Intel Software Guard Extensions (SGX) [32], [33] or Arm TrustZone [34] in mobile or IoT devices, opens exciting new opportunities to build secure yet efficient systems. Infact, TEEs allow near-to-metal execution speed while offering several additional security guarantees, including local and remote attestation (*e.g.*, the ability to verify the authenticity of the executable code as well as the execution environment), integrity protection, etc.

Isolated *enclaves* protect the code and the data from malicious users, operating systems, cloud providers and in general from any privileged user with administrative or even physical access to the processor. The trusted computing base (TCB) is basically reduced to the CPU die and the CPU manufacturer. Enclaves are limited to a subset of the available memory on a machine (the *enclave page cache*), with the most recent



(a) Training data = 100%

(b) Training data = 50%

Fig. 7: Test accuracy of the proposed learning system on CIFAR-10

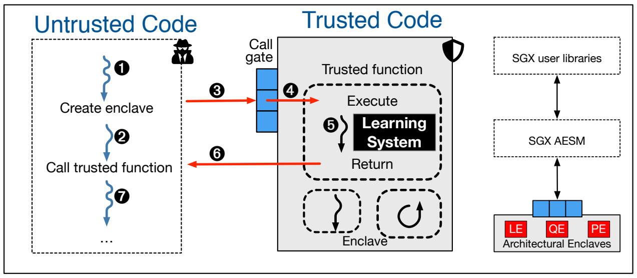


Fig. 8: TEE workflow: a trusted SGX enclave shields the learning system from malicious attackers.

server-grade processors offering up to 256 MB of encrypted memory [35]. Figure 8 depicts the typical execution workflow with Intel SGX.

Initial attempts started to emerge to exploit TEEs to efficiently execute complete machine-learning pipelines into enclaves [36]. In the remainder, we describe our vision toward new research challenges that TEEs offer in the development, deployment, and execution of shielded learning networks resilient.

Fig. 8 shows how the learning system must be designed to execute inside an SGX enclave. While this sounds straightforward, it entails several challenges.

For instance, enclaves cannot execute system calls, delegating their execution to the external, untrusted environment. This becomes problematic for instance when dealing with a distributed file-system (*e.g.*, HDFS).

Another challenge is when dealing with the very limited memory available inside the enclaves. One needs to carefully design the learning system to move into the trusted environment only the portions of data used by the current processing step.

Finally, there is a great diversity among the various TEE currently available, for instance on mobile or server-grade machines. One design for a trusted learning system might not reflect in the best conditions across the different configurations along several dimensions, *e.g.*, security, computing or energy efficiency. Since frameworks to transparently adapt a given design to different TEEs are still in their infancy (*i.e.*, Google Asylo [37]), we believe that additional considerations are required toward a truly secure cross-TEE learning framework.

VI. Conclusion

Motivated by the significant presence of dirty labels and their detrimental impacts on machine learning based solutions, we first propose a visionary learning framework that can robustly filter and cleanse dirty labels for both regular and deep machine learning models. The core of proposed framework is to combine artificial and human intelligence via three sequential strategies: (i) quality model to filter the data, (ii) active learning strategies from human experts, and (iii) imitating the process of experts' label correction. Using various of image benchmarks with different percentages of corrupted labels, we show that such a collaborative learning framework can not only ensure the learning accuracy but also accelerate the learning efficiency by focusing on subset set of informative data. Finally, we extend the robust perspective from obtaining trusted data to leveraging trusted exeecution for machine learning sytems.

While our initial results present promising directions in combating the challenging problem of dirty labels, numerous practical aspects are yet to be considered. First of all, the robustness issues can arise from both data as well as the execution environment. The security risk of computing platforms increases by many folds in recent years. It becomes imperatively important to use the trusted data as well as trusted execution environment. Secondly, increasingly number of machine learning models are trained distributively on data that are continuously sensed and collected. This hence calls for the distributed learning framework that can decentralizedly cleanse and filter the data. Moreover, in addition to erroneous annotation in the process of data collection, other sources of label corruptions could arise from the poison attacks, meaning dirty labels are maliciously injected into the data. Such types of dirty labels significantly increase the difficulty of training robust machine learning models and invite novel and practical solutions.

Acknowledgment

This work has been partly funded by the Swiss National Science Foundation NRP75 project, Dapprox, 407540_167266.

References

[1] Y. Yan, R. Rosales, G. Fung, R. Subramanian, and J. Dy, "Learning from multiple annotators with varying expertise," *Machine learning*, vol. 95, no. 3, pp. 291–327, 2014.

[2] A. Blum, A. Kalai, and H. Wasserman, "Noise-tolerant learning, the parity problem, and the statistical query model," *Journal of the ACM (JACM)*, vol. 50, no. 4, pp. 506–519, 2003.

[3] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *International Conference on Learning Representations*, 2014. [Online]. Available: http://arxiv.org/abs/1312.6199

[4] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*, 2016.

[5] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6572

[6] Y. Wang, W. Liu, X. Ma, J. Bailey, H. Zha, L. Song, and S.-T. Xia, "Iterative learning with open-set noisy labels," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[7] S. E. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, "Training deep neural networks on noisy labels with bootstrapping," in *3rd International Conference on Learning Representations, ICLR, Workshop Track Proceedings*, 2015.

[8] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," in *Proceedings of the 35th International Conference on Machine Learning, ICML*, 2018, pp. 4331–4340.

[9] I. Jindal, M. Nokleby, and X. Chen, "Learning deep networks from noisy labels with dropout regularization," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 2016, pp. 967–972.

[10] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *5th International Conference on Learning Representations, ICLR*, 2017.

[11] E. Malach and S. Shalev-Shwartz, "Decoupling" when to update" from" how to update"," in *Advances in Neural Information Processing Systems*, 2017, pp. 960–970.

[12] 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 *Advances in Neural Information Processing Systems*, 2018, pp. 8527–8537.

[13] Y. Wang, X. Ma, M. E. Houle, S.-T. Xia, and J. Bailey, "Dimensionality-driven learning with noisy labels," *International Conference on Machine Learning (ICML)*, 2018.

[14] Z. Zhao, S. Cerf, R. Birke, B. Robu, S. Bouchenak, S. Ben Mokhtar, and L. Y. Chen, "Robust anomaly detection on unreliable data," in *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, June 2019, pp. 630–637.

[15] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)." [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[16] ——, "Cifar-100 (canadian institute for advanced research)." [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[17] H. Ng and S. Winkler, "A data-driven approach to cleaning large face datasets," in *2014 IEEE International Conference on Image Processing (ICIP)*, Oct 2014, pp. 343–347.

[18] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

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

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1409.1556

[21] Z. Zhao, R. Birke, R. Han, B. Robu, S. Bouchenak, S. Ben Mokhtar, and L. Y. Chen, "RAD: On-line Anomaly Detection for Highly Unreliable Data," *arXiv e-prints*, p. arXiv:1911.04383, Nov 2019.

[22] C. M. Bishop, *Pattern recognition and machine learning*. Springer Science+ Business Media, 2006.

[23] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[25] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[26] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1944–1952.

[27] "Azure Machine Learning," https://azure.microsoft.com/en-us/services/machine-learning/, 2019.

[28] "Google AI Platform," https://cloud.google.com/ai-platform/, 2019.

[29] "AWS on ML," https://aws.amazon.com/machine-learning/, 2019.

[30] T. Graepel, K. Lauter, and M. Naehrig, "Ml confidential: Machine learning on encrypted data," in *International Conference on Information Security and Cryptology*. Springer, 2012, pp. 1–21.

[31] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser, "Machine learning classification over encrypted data." in *NDSS*, vol. 4324, 2015, p. 4325.

[32] V. Costan and S. Devadas, "Intel SGX Explained." *IACR Cryptology ePrint Archive*, vol. 2016, no. 086, pp. 1–118, 2016.

[33] T. Dinh Ngoc, B. Bui, S. Bitchebe, A. Tchana, V. Schiavoni, P. Felber, and D. Hagimont, "Everything You Should Know About Intel SGX Performance on Virtualized Systems," *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 3, no. 1, pp. 5:1–5:21, Mar. 2019. [Online]. Available: http://doi.acm.org/10.1145/3322205.3311076

[34] J. Amacher and V. Schiavoni, "On the Performance of ARM TrustZone," in *IFIP International Conference on Distributed Applications and Interoperable Systems*. Springer, 2019, pp. 133–151.

[35] "Intel Xeon E-2200 Processor," https://intel.ly/2KmE7sL.

[36] R. Kunkel, D. L. Quoc, F. Gregor, S. Arnautov, P. Bhatotia, and C. Fetzer, "TensorSCONE: A Secure TensorFlow Framework using Intel SGX," *arXiv preprint arXiv:1902.04413*, 2019.

[37] "Google Asylo," https://asylo.dev/, 2019.