The Fault in Our Data Stars: Studying Mitigation Techniques against Faulty Training Data in Machine Learning Applications

Abraham Chan, Arpan Gujarati, Karthik Pattabiraman, Sathish Gopalakrishnan
The University of British Columbia (UBC), Vancouver, BC, Canada
Email: abrahamc@ece.ubc.ca, arpanbg@cs.ubc.ca, {karthikp, sathish}@ece.ubc.ca

Abstract—**Machine learning (ML) has been adopted in many safety-critical applications like automated driving and medical diagnosis. Incorrect decisions by ML models can lead to catastrophic consequences, such as vehicle crashes and inappropriate medical procedures, thereby endangering our lives. The correct behaviour of a ML model is contingent upon the availability of well-labelled training data. However, obtaining large and high-quality training datasets for safety-critical applications is difficult, often resulting in the use of faulty training data.**

We compare the efficacy of five different error mitigation techniques, derived from a survey of more than 200 related articles, which are designed to tolerate noisy/faulty training data. We experimentally find that the error mitigation capabilities of these techniques vary across datasets, ML models, and different kinds of faults. We further find that *ensemble learning offers the highest resilience* among all the techniques across different configurations, followed by label smoothing.

Index Terms—**Error resilience, Machine learning, Training**

I. Introduction

Machine learning (ML) applications are actively deployed in many safety-critical domains including medical diagnosis [1] and autonomous vehicles (AVs) [2]. For example, deep neural networks (DNNs) have been developed and trained to automatically and efficiently screen patients with COVID-19 based on chest X-ray images [3]. Incorrect inferences could cause either unnecessary medical procedures or a neglect of care, which could have serious health consequences to patients [4, 5]. Similarly, incorrect inferences by an AV could cause collisions, endangering our lives and property.

Many of these applications use *supervised learning*, where labelled examples are collected *a priori* for training the ML components. The training data, thus, has a significant influence on the correctness of ML applications (we focus on classification applications). *We examine different approaches to mitigating the effects of these faults in training data sets.* Our goal is to help developers choose the best technique for protecting their ML models from training data faults.

A. Motivation

To generate large amounts of training data, which is required to achieve high classification accuracy, ML developers widely rely on crowd-sourcing and automatic labelling techniques [6]. In specialized domains such as medical sciences, this can be a monumental challenge because of logistical and legal barriers [7]. Furthermore, it is expensive to have multiple experts manually validate every image-label pair in a large training dataset. Hence, despite best efforts, training data may be incomplete or contain mislabelled entries.

For example, ChestX-ray14 [8] is a large-scale medical dataset with more than $100,000$ images, spanning 14 types of diseases. Tang *et al.* [9] found that 20% of the images in a random sample of this dataset were mislabelled. The popular open source Udacity Dataset 2, used for training AVs, has 33% of the images that are either mislabelled or are missing labels [10]. Even large open datasets like ImageNet [11] have been found to contain mislabelled data [12].

There have been many techniques proposed to mitigate, or compensate for, faulty training data in ML models [13]. These techniques are, however, not directly comparable as they often use different metrics and datasets. Moreover, many of these techniques require considerable tuning or adaptation to get them to work on ML architectures and datasets. On the other hand, there exist techniques [14, 15] that defend against data poisoning attacks, where the attacker purposely crafts training data faults so that specific input targets are misclassified. These defense techniques also try to identify and clean mislabelled data during training. However, they often use pattern matching on deep features to identify maliciously mislabelled data, which is ineffective against more general faults in the training data. We seek to answer the following question that remains open (to the best of our knowledge): *How to select a technique to best protect ML models against the effects of faulty training data? Note that we do not study techniques against data poisoning attacks in this paper.*

B. Contributions

We conducted a survey of about 200 research papers published between 2017 and 2021 on protecting ML models from faulty training data. We shortlisted 50 papers with well-documented techniques. We divided these into five *training-data fault mitigation* (TDFM) approaches: (1) label smoothing [16], (2) label correction [17], (3) robust loss [18], (4) knowledge distillation [19], and (5) ensembles [20]. For each approach, we selected a representative implementation that is generic across datasets or model architectures. Finally, we injected training data faults, and compare these five approaches with each other in terms of their ability to tolerate the training data faults injected by us. Notably, we used the same metrics and datasets, so as to obtain an "apples-to-apples" comparison.

We consider three types of faults in training data [21],

1) *mislabelling faults* - data is erroneously labelled,
2) *repetition faults* - input-output pairs are repeated,
3) *removal faults* - a fraction of data may be deleted.

We choose these three fault types in part because they were found to be the most significant fault types in a well-used music dataset, GTZAN [22]. These types of fault are also rampant in datasets covering safety-critical domains [23].

We evaluate the effectiveness of the five TDFM approaches (identified from our survey) using seven commonly used ML models, and by injecting the training data of each model with mislabelling, repetition, and removal faults before applying the TDFM techniques. We also consider three different datasets, and use metrics beyond just accuracy, capturing the differences in resilience between protected and non-protected models [21].

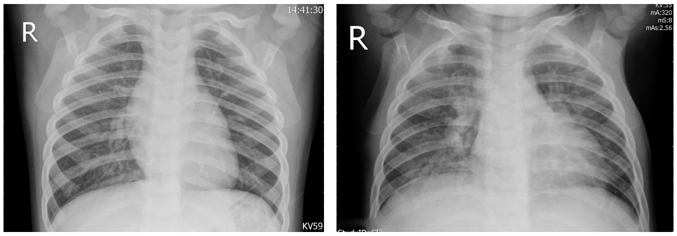
There are three main findings in our study. First, some of the commonly used TDFM techniques may not be effective across all the ML models. Second, we find that the efficacies of TDFM techniques vary across different datasets and fault types. Finally, we find that of all the TDFM techniques, *ensembles offer the highest resilience across all models, fault types, and datasets,* but incur high overheads. *Label smoothing* comes a close second, but does not incur as much overhead.

II. Motivating Example

What happens when faults are present in a training dataset? We present an example from the health sciences. We consider a publicly available Pneumonia dataset [24], which consists of 5,863 X-ray images of pediatric patients in the Guangzhou Women and Children's Medical Center in China. Unlike many other medical datasets, every image and label here is screened by two expert physicians, making this a well-curated dataset.

We train a ResNet50 model on the Pneumonia dataset, obtaining an accuracy of 90%; we refer to this trained model as the *golden model*. We then inject 10% mislabelling faults into the training data (uniformly at random). This mislabelling rate is in line with average estimates (*i.e.*, 7.4% to 20%) of mislabelled images in publicly available medical datasets [9, 25, 26]. When ResNet50 is trained on the mislabelled training dataset, we obtain an accuracy of only 55%, a significant drop in classification accuracy—a patient's chance of a correct diagnosis is only slightly better than tossing an unbiased coin!

We can examine two specific images from the dataset (Fig. 1). The golden model correctly classifies these images as *(a) normal* and *(b) pneumonia*, respectively. However, the ResNet50 model trained on mislabelled data (we call this the *faulty model*) classifies both these images incorrectly. It classifies the normal X-ray of a healthy patient (Fig. 1a) as having pneumonia, but classifies the X-ray image with pneumonia (Fig. 1b) as normal. Thus, while a healthy patient could be subjected to additional screening or be prescribed unneeded medication, a pneumonia patient may be left untreated, leading to further respiratory complications or even a lung failure. This example demonstrates that even small amounts of faults in the training data can have significant impact on the functional correctness of safety-critical systems. Therefore, it is important to mitigate the faults using TDFM techniques.



(a) Normal X-ray (b) X-ray with pneumonia

Fig. 1: With 10% mislabellings, ResNet50 classifies these images from the Pneumonia dataset incorrectly. It labels (a) as the one with Pneumonia, whereas it labels (b) as normal.

III. Methodology

We first explain our methodology to select the different TDFM techniques, followed by an explanation of the chosen techniques. We then introduce the metrics for evaluation, and finally apply them to the motivating example in Section III-D.

A. Survey Technique

We take a two-pronged approach to select a representative set of TDFM approaches. We perform a detailed analysis of related survey papers [13, 37–39]. In addition, we also aggregate results from three sources, namely IEEE Xplore, arXiv, and Github, using search keywords "label noise" and "noisy training". Overall, we identify more than 200 relevant research articles. Among these, we focus on articles that propose techniques to mitigate the effects of label noise, rather than mechanisms for detecting label noise. We focus our search on techniques tolerating mislabelling in neural networks[1] as we found no adequate techniques for tolerating data removal, and no techniques at all for repetition. For example, current approaches [40, 41] against data removal have only been implemented as specific layer-wise architectural modifications to shallow neural networks. We reject techniques that have been outperformed by more recent work, as well as techniques applied to ML problems other than image classification (as all our datasets are image classification tasks).

The above criteria narrow down the articles to about 50. We categorize the articles into five TDFM approaches: label smoothing, label correction, robust loss, knowledge distillation, and ensembles. Finally, we select one representative technique for each approach. We consider a technique to be representative of a TDFM approach if (1) its code is available and easily modifiable to include new neural networks and datasets, (2) it has been evaluated on more than one neural network architecture type and dataset, (3) it is capable of tolerating artificial noise, (4) it does not rely on pre-trained weights, and (5) is not on a combination of other techniques.

[1] We consider neural networks in this work as they are the dominant ML models used for image classification tasks.

TABLE I: Top three techniques for the five TDFM approaches. Representative techniques are marked with an asterisk.

TDFM Approach	Technique	Code?	Architecture-Agnostic?	Artificial Noise?	Not Pre-Trained?	Standalone?
Label Smoothing	Label Relaxation* [16]	✓	✓	✓	✓	✓
	Lukasik et al. [27]	✗	✗	✓	✓	✗
	OLS [28]	✗	✓	✓	✓	✓
Label Correction	Meta Label Correction* [17]	✓	✓	✓	✓	✓
	ProSelfLC [29]	✗	✗	✓	✓	✓
	SMP [30]	✓	✗	✗	✗	✓
Robust Loss	Active-Passive Losses* [18]	✓	✓	✓	✓	✓
	Charoenphakdee et al. [31]	✓	✗	✓	✓	✓
	Zhang et al. [32]	✓	✗	✓	✓	✓
Knowledge Distillation	CMD-P [33]	✗	✓	✓	✗	✓
	KD-Lib [34]	✓	✓	✗	✓	✗
	Self Distillation [19]	✓	✓	✗	✓	✓
Ensemble	LTEC [35]	✓	✗	✓	✓	✓
	SELF [36]	✗	✗	✓	✓	✗
	Super-Learner [20]	✗	✓	✗	✓	✓

The above criteria are needed as our goal is to have an apples-to-apples comparison of all techniques on identical datasets with fault-injected training data. The first criterion is important since we want to evaluate each technique on our own datasets. Secondly, some articles present their results for only one type of neural network architecture (*e.g.*, they may present results for only the ResNet family). These results may not extend to other types of architectures like VGG or MobileNet. Thirdly, some techniques may be designed for inherently noisy datasets, *e.g.*, Food-101 [42]. Such techniques are of little use as well because they assume a pre-determined distribution of noise and focus on tolerating noise from only certain label classes. Fourthly, many techniques also rely on pre-trained weights (*e.g.*, ImageNet [11] weights) as their starting point. However, these techniques cannot be easily compared with other techniques, and pre-trained weights may even hinder the performance of ML models [43]. Finally, for a fair comparison, we require that the techniques selected for evaluation are standalone and not a combination of other TDFM techniques.

Table I summarizes our selection process. It shows the top three articles for each TDFM approach, along with how they satisfy our selection criteria. Techniques that meet all selection criteria are emphasized using an asterisk (*); we select these as the sole representative of the respective TDFM approach. For Knowledge Distillation and Ensemble approaches, we could not identify a single representative technique that satisfies all criteria. For these approaches, we re-implemented representative techniques in TensorFlow [44] using the descriptions and configurations provided in the respective top three articles.

B. Background: TDFM Approaches Chosen

1) Label Smoothing (LS): Labels (outputs) in multi-class datasets are often represented using one-hot encoding, *e.g.*, in a dataset with three label types, vector $[0, 1, 0]$ can encode an output of the second label type. Such "hard labels" result in steep gradients during backpropagation. *Label smoothing* allows associating non-zero probabilities with each label type, reducing this gradient. That is, if K denotes the number of label types and p_i denotes a hard label probability (0 or 1), the smoothened output probability for label type i is $q_i = (1 - \alpha)p_i + \alpha(1/K)$, where α is a hyperparameter. For example, $\alpha = 0.1$ transforms one-hot encoding $x = [0, 1, 0]$ into $y = [0.033, 0.933, 0.033]$. *Label relaxation* [45] is an extension of label smoothing - label smoothing assumes a uniform distribution over all non-target label types, as indicated by coefficient $1/K$, whereas label relaxation generalizes this by allowing the model to choose from any distribution, *e.g.*, resulting in $z = [0.044, 0.933, 0.022]$. Label relaxation mitigates the effect of mislabelled data by reducing the distance between correct and incorrect encodings.

2) Label Correction (LC): This is a meta-learning approach that attempts to correct faulty labels in the training data during training. Two neural networks are simultaneously trained – the primary model for the actual classification task and a secondary model to identify and correct faulty labels. The secondary model must be trained on a clean subset of the main dataset. A clean subset is obtained by manually verifying a proportion of the training data. For artificial noise injection experiments, a clean subset is formed by reserving a portion of the training data from fault injection. The fraction of dataset used as clean data is a hyperparameter, denoted γ.

3) Robust Loss (RL): During training, *loss functions* determine the deviation between predicted and actual labels in every iteration. Cross Entropy (CE) [46] is the most common loss function used but is not robust to label noise [47], whereas an Active-Passive Loss (APL) function is more robust [18]. APL is defined as a weighted sum of two loss functions using hyperparameters α and β, *i.e.*, $\mathcal{L}_{APL} = \alpha \cdot \mathcal{L}_{Active} + \beta \cdot \mathcal{L}_{Passive}$. $\mathcal{L}_{Active}$ minimizes the loss on the target class but also reduces the overall accuracy; $\mathcal{L}_{Passive}$ minimizes the loss on non-target classes and reduces the underfitting introduced by the active loss function. We use Normalized Cross Entropy (NCE) as $\mathcal{L}_{Active}$ and Reverse Cross Entropy (RCE) as $\mathcal{L}_{Passive}$, both of which are robust to label noise unlike CE [18].

4) Knowledge Distillation (KD): This involves training two models, a teacher model and a student model with knowledge

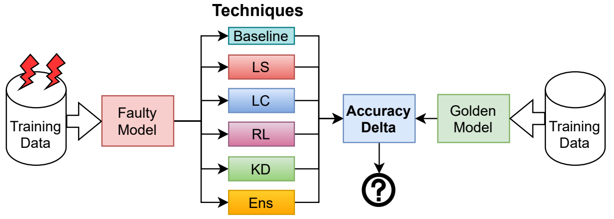


Fig. 2: Measuring AD across TDFM techniques between faulty and golden models.

from the teacher model [48]. The teacher model's output activation function is modified to distill information.[2] The student model is trained with a combination of two loss functions, its own and the teacher's distilled softmax; their relative weight is controlled by hyperparameter α – a larger α gives more weight to new information over previously acquired knowledge. This results in the student achieving a higher accuracy than its teacher. The teacher model is usually deeper than (*i.e.* has more layers) the student model. However, we use *self distillation*, where the teacher and student models are the same, which has been found even more effective [19].

5) Ensemble Learning (Ens): This approach involves training multiple models, and then combining their outputs at inference time using simple majority voting. Since individual models in an ensemble learn sufficiently diverse aspects of the feature space, the ensemble can tolerate the effects of faulty training data as demonstrated in our prior work [21]. The number of models in an ensemble is a hyperparameter, n.

C. Measuring Reliability

We measure the accuracy and the accuracy delta (AD) of the models. The AD is the proportion of test images that are misclassified by the faulty model out of all test images that were correctly classified by the golden model. *A more resilient model has a lower AD.* We find that the proportion of test images misclassified by the golden model, but correctly classified by the faulty model to not be significant. The AD measures the precise effect of faulty training data on the model's outcome, by not double-counting test images that are misclassified by both the golden and faulty models, and thereby enables fair comparison between the different models.

We demonstrate how we measure the AD for each technique (Fig. 2). First, the *golden model* refers to the model, trained on training data without faults. Secondly, the *faulty model* refers to the same model, trained on faulty training data. The *baseline* refers to the faulty model without any TDFM techniques applied. Finally, we measure the AD of each TDFM technique applied on the faulty model relative to the golden model.

TABLE II: Image classification datasets used

Name	**Dataset Size**		**Task (# Classes)**
	Training	**Test**	
CIFAR-10 [49]	50,000	10,000	Objects and animals (10)
GTSRB [50]	39,209	12,630	Traffic signs (43)
Pneumonia [24]	5,239	624	Chest X-rays (2)

TABLE III: Neural network architectures used

Name	**Depth**	**Architecture Summary**
ConvNet	Moderate	3 Conv + 3 FC + Max Pooling
DeconvNet	Moderate	4 Conv + 2 FC w/ 0.5 Dropout
VGG11	Deep	13 Conv + 3 FC + Max Pooling
VGG16	Deep	13 Conv + 3 FC + Max Pooling
ResNet18	Deep	17 Conv + 1 FC + Avg Pooling
MobileNet	Deep	27 Conv + 1 FC + Avg Pooling
ResNet50	Deep	49 Conv + 1 FC + Avg Pooling

D. Example

We revisit our motivating example of mislabelling in the Pneumonia dataset (Section II) and apply each of the TDFM techniques independently on the faulty ResNet50 model, resulting in 5% AD (LS), 29% AD (LC), 15% AD (RL), 13% AD (KD), and 5% AD (Ens). Label smoothing and Ensemble learning yield the lowest AD. We thus conclude that these two TDFM techniques are the most resilient for 10% mislabelling in Pneumonia. We would like to know if *certain techniques outperform others for all models, fault types, and datasets? If not, how should we identify a suitable TDFM technique?*

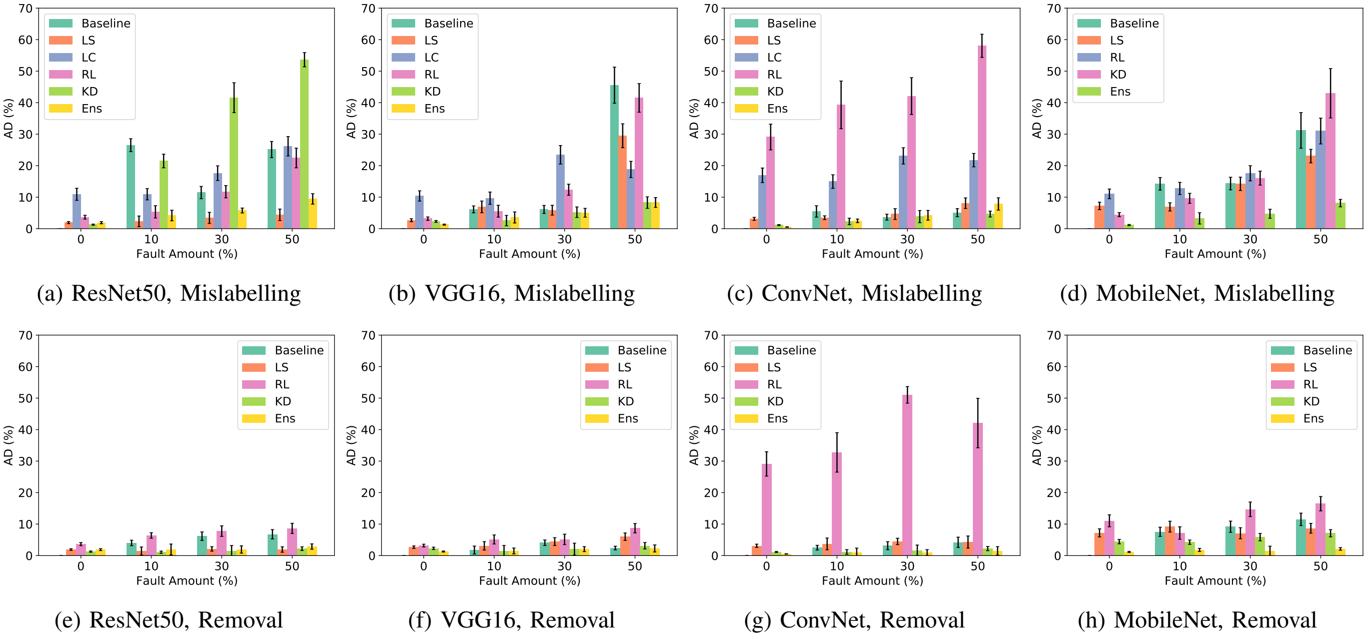
IV. Evaluation

Our goal is to measure the AD of each technique and compare to the baseline. We use three datasets for our evaluation (Table II): GTSRB, Pneumonia, and CIFAR-10. GTSRB and Pneumonia both represent safety-critical problems, namely road sign recognition for self-driving cars and medical diagnosis, respectively. We also use CIFAR-10 due to its balanced nature (*i.e.*, equal number of images per label class) and as it represents a general object detection problem. Pneumonia is much smaller in size than other datasets, reflecting the difficulty of obtaining quality medical images for training [7].

All three datasets we used are well-curated. For example, images in CIFAR-10 were manually labelled [49] and images in the GTSRB dataset were labelled automatically but verified by humans [50]. All images in the Pneumonia dataset were manually verified by two expert physicians. Therefore, we assume there are no inherent training faults in these datasets, and that the only faults are those that we inject into the training data. The golden model is hence trained on the original dataset. We considered only image data as it was difficult to find sufficiently well-curated data sets for other applications.

For our experiments, we used seven popular neural networks for image classification, as shown in Table III, of varying architecture types and depth: ConvNet, DeconvNet, MobileNet, ResNet18, ResNet50, VGG11, and VGG16. We believe that the diversity of the neural network architectures encourages models to learn different features from a common dataset.

[2] For example, consider *softmax*: given the number of classes K and hyperparameter T, the softmax activation function $\exp(z_i/T)/\sum_j^K \exp(z_j/T)$ converts the raw model outputs (*logits*) z_i and z_j at indices i and j into probabilities that determine the likelihood with which the input belongs to each label class. Regular softmax sets $T = 1$, producing a sharp probability distribution around a single predicted class. A distilled softmax instead sets $T > 1$, enabling softer distributions (similar to label smoothing).



(a) ResNet50, Mislabelling (b) VGG16, Mislabelling (c) ConvNet, Mislabelling (d) MobileNet, Mislabelling

(e) ResNet50, Removal (f) VGG16, Removal (g) ConvNet, Removal (h) MobileNet, Removal

Fig. 3: AD of models protected with TDFM techniques versus the baseline, trained on GTSRB, with mislabelling faults in (a-d) and removal faults in (e-h). The error bars in the results indicate the 95% confidence intervals. Lower values are better.

Each TDFM technique requires certain hyperparameter choices (such as α and β for Robust Loss (Section III-B3)). We used the hyperparameters recommended by the implementers of the techniques [16–18, 48]. We had found ensembles of size 5 models to be most effective in our prior work [21]. *Our ensemble consists of the 5 models with the lowest baseline* AD: ConvNet, MobileNet, ResNet18, VGG11, and VGG16.

We use the TF-DM fault injector [51] for injecting faults into training datasets. We inject the three types of training data faults described earlier (Section I). We first train each model with fault-free training data to obtain a golden model, and then train the same model, applying each TDFM technique, with fault injected training data, resulting in a faulty model. We then collect the prediction results on the test dataset for both the golden and faulty models, and calculate their AD. We consider three different fault percentages: 10, 30 and 50 for each fault type, to study the effects of different fault amounts. For example, a fault percentage of 30% mislabelling means that 30% of the training data was mislabelled (at random).

The average training time for each configuration was about 45 minutes. To reduce the variance in our results, we evaluated each configuration 20 times. In total, training took 33 days of computational time, and inference took 6 hours, running on Nvidia P100 GPUs with an Intel E5-2650-v4-Broadwell CPU.

A. *Baseline Accuracy without Fault Injection*

We first measure the classification accuracies of each model, trained without any injected training data faults, for each TDFM technique applied. It is important to understand the effects of each technique on the golden model accuracy before measuring the AD after fault injection. We show the results only for four models due to space constraints (Table IV), but find similar results in other models of the same architecture class (*i.e.*, ResNet50 and ResNet18). However, as we were not able to run label correction on MobileNet, we skipped it.

We observe that the TDFM techniques do not affect the golden model accuracy in most cases. Exceptions include label correction and robust loss, which have degraded accuracy on models trained with the Pneumonia dataset, as these techniques require larger sized datasets to be more effective. A small sized dataset does not provide sufficient samples to train the secondary model in label correction. For robust loss, it applies the NCE+RCE loss function (Section III-B), where the active part of the loss function is prone to underfitting. The small size of the training dataset causes severe underfitting, which is not recoverable by the passive part of the loss function.

Knowledge distillation has high baseline accuracy, in fact, the highest across models for GTSRB. However, it is not the most resilient TDFM technique (shown later in Section IV-B).

B. AD *across Models*

First, we analyze the effectiveness of the different TDFM techniques across the seven models. Due to space constraints, we show the results for four models with mislabelling faults for the GTSRB dataset (Figs. 3a to 3d); the other configurations exhibit similar results and are hence not shown.

We observe in most configurations, the baseline's AD is higher than the AD of the models after applying the TDFM techniques (recall that higher AD means it is less resilient). This is expected as the goal of TDFM techniques is to tolerate faults in the training data. However, their effectiveness varies as evidenced by some techniques having lower ADs than others and hence being more effective. For these configurations, we find *both label smoothing and ensembles to be highly*

TABLE IV: Model accuracies when trained without fault injection. Datasets: CIFAR-10 (1), GTSRB (2), Pneumonia (3). The highest accuracy for each configuration is emphasized.

Model	Dataset	Base	LS	LC	RL	KD	Ens
ResNet50	1	**93%**	**93%**	91%	86%	73%	85%
	2	91%	96%	86%	94%	**97%**	96%
	3	90%	91%	78%	74%	88%	**93%**
VGG16	1	88%	**93%**	**93%**	82%	89%	85%
	2	93%	94%	87%	95%	**96%**	**96%**
	3	90%	86%	73%	77%	85%	**93%**
ConvNet	1	82%	76%	82%	81%	80%	**85%**
	2	93%	92%	79%	77%	**97%**	96%
	3	92%	88%	75%	68%	91%	**93%**
MobileNet	1	**87%**	84%	-	73%	73%	85%
	2	88%	87%	-	86%	92%	**96%**
	3	91%	90%	-	76%	90%	**93%**

effective, knowledge distillation having a mixed effect, while robust loss and label correction are largely ineffective.

We find that ensembles consistently outperform the individual models equipped with TDFM techniques. This is because the models used to construct the ensembles have different architecture types (*e.g.* residual layers in ResNet models, stacked convolutional layers in VGG models), the ensemble can tolerate faults provided the majority of the individual models do not misclassify simultaneously.

Label smoothing is able to mitigate the effects of mislabelled data without inhibiting the models' ability to learn or being affected by network architectures. However, it is not as effective as ensembles at mitigating training data faults.

TDFM techniques do not always improve resilience over the baseline. For instance, in Fig. 3a, knowledge distillation has lower AD than the baseline when the mislabelling is 10% or less, while having a higher AD when mislabelling is 30% or more. When mislabelling rates are low, knowledge distillation functions as a learned label smoothing function (Section III-B). However, when mislabelling rates are high, the student model is unable to filter out incorrect information from the teacher, resulting in a higher AD, *i.e.*, a "garbage in, garbage out" scenario [33]. We also observe that some models with knowledge distillation, despite having the highest baseline accuracies, were not the most resilient. For example, knowledge distillation in ResNet50 had a baseline accuracy of 97% for GTSRB (Table IV), but had 42% and 55% AD at 30% and 50% mislabelling respectively (Fig. 3a).

Further, robust loss and label correction have a higher AD than the baseline (Fig. 3c). Compared with other models in our experiment, ConvNet is a shallow model, with fewer layers. We find that robust loss performs poorly for shallow models, which are unable to learn deep features of training data. This is because robust loss attempts to reduce overfitting to mislabelled data by using softer (*i.e.*, normalized) loss functions. However, the softened loss functions inhibit the ability of shallower models to learn from the data.

Label correction utilizes two models that learn concurrently, and the secondary model acts as an additional validation loss function to the primary model, resulting in a softer loss function like robust loss. In contrast, the baselines use the popular cross-entropy loss function, which does not suffer from this effect. The low resilience of both label correction and robust loss on models like ConvNet show that alternative loss functions have a negative effect on shallower models, but have a positive effect on deeper networks (*e.g.* ResNet50).

Observation 1 *Ensembles and label smoothing are the most resilient TDFM techniques across the ML models.*

C. AD *across Fault Types*

We expand our evaluation to other fault types such as data removal, while continuing to use the GTSRB dataset (Figs. 3e to 3h). We do not run label correction on fault types other than mislabelling since label correction has no effect on them. We make two observations. First, all models have a lower AD compared to mislabelling faults. This means that most models can still learn effectively with fewer training examples (as much as 50% removals) in this dataset. Secondly, we see that the baseline AD is still reduced by most techniques. In fact, the TDFM techniques effective against mislabelling were also effective against removal faults, except for robust loss on ConvNet (Fig. 3g). Robust loss is ineffective against data removal faults on ConvNet for reasons that are similar to those explained earlier (Section IV-B). We do not show results for data repetition because the trends are similar to data removal.

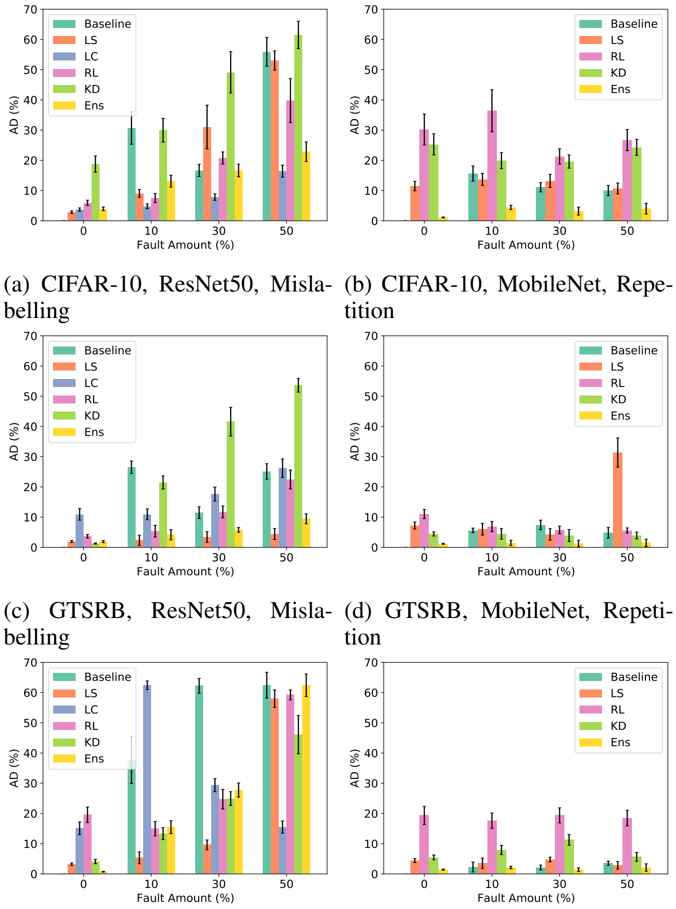
We also performed fault injection using combinations of multiple fault types (*e.g.*, mislabelling combined with removal. However, we did not find significant differences with the results of injecting individual fault types. For example, when we combined mislabelling with either removals or repetitions, the AD was statistically similar to that of mislabelling only. Similarly, when we combine removal with repetition, the AD was statistically similar to that of repetition only. Therefore, we believe that our findings are applicable even under multiple training data faults.

Observation 2 *TDFM techniques effective against data mislabelling are also effective against removal and repetition.*

D. AD *across Datasets*

We expand our evaluation to all three datasets (Fig. 4). We observe that CIFAR-10 and Pneumonia generally have higher AD than GTSRB. While CIFAR-10 contains training images from fewer image classes than GTSRB, CIFAR-10's images often contain multiple objects in the image background while GTSRB's images are more focused on the road sign. Pneumonia is about $1/10$-th the size of the other two datasets. Since ML models usually require lots of training data for accuracy, we expected models trained with Pneumonia to be less resilient overall. Surprisingly however, the models were quite resilient to both removal and repetition faults across all datasets, (Figs. 4b, 4d and 4f) including Pneumonia.

Further, we observe that some techniques are effective only on specific configurations. *We, however, find that ensembles are resilient across most configurations of models, fault types,*



(a) CIFAR-10, ResNet50, Mislabelling

(b) CIFAR-10, MobileNet, Repetition

(c) GTSRB, ResNet50, Mislabelling

(d) GTSRB, MobileNet, Repetition

(e) Pneumonia, ResNet50, Mislabelling

(f) Pneumonia, MobileNet, Repetition

Fig. 4: AD of individual models, compared with models protected with TDFM techniques, when trained with faulty CIFAR-10, GTSRB and Pneumonia datasets. The error bars indicate the 95% confidence intervals. Lower values are better.

and datasets. Label smoothing also improves resilience across most configurations, though not as much as ensembles do.

Label correction had a mixed effect, working well for CIFAR-10 and Pneumonia, while underperforming for GTSRB. Notice (Fig. 4a and Fig. 4e) that label correction has the lowest AD among all techniques when 50% mislabelling is injected into CIFAR-10 and Pneumonia, but has the second highest AD for GTSRB for the same configuration (Fig. 4c). CIFAR-10 and Pneumonia have fewer classes, 10 and 2 respectively, than GTSRB, which has 43 classes. Label correction uses a secondary model to correct labels - we found that the number of classes in the dataset had an impact on the secondary model's ability to correct labels. Because the secondary model uses a multilayer perceptron, it cannot handle larger numbers of classes. Surprisingly however, the size of the dataset did not have an impact on label correction - we had expected the secondary model to have fewer clean samples from a smaller dataset, hence, hindering label correction.

Similar to GTSRB, in other datasets, knowledge distillation has a higher AD than the baseline when mislabelling increases, but a lower AD at lower fault amounts. For repetition faults, however, knowledge distillation has the second highest AD after robust loss (Figs. 4b, 4d and 4f). By default, more weight is given to the teacher's distilled loss. When the student encounters repeated data, the weight is implicitly shifted towards the student, reducing the technique's effectiveness.

Robust loss performed well when mislabelling was 30% or less across datasets, but performed poorly at 50% mislabelling. For repetition faults, robust loss had even higher AD in CIFAR-10 (Fig. 4b) and Pneumonia (Fig. 4f) than for GTSRB (Fig. 4d). Finally, robust loss has a high AD for most configurations in the Pneumonia dataset (Figs. 4e and 4f).

Observation 3 *Ensembles are resilient across different models, fault types, and datasets. Label smoothing comes second.*

E. Runtime Overhead Analysis

We measured the runtime overhead of the TDFM techniques, consisting of the training and inference overheads respectively. For inference time, we find that the training overhead is 1×, meaning no change, across all configurations except for ensembles, which have a 5× inference time overhead (as they consist of five models). For training time, there is more variation across techniques and datasets. Label smoothing has the lowest training overhead since it is applied only once to the training dataset, prior to backpropagation. Knowledge distillation involves training the model twice, through the teacher and the student models. However, instead of a 2× overhead, it only incurs a 1.5× overhead overall, since the student model trains faster than the parent. Robust loss has varying training overheads, as certain configurations behave differently to loss functions. Label correction has a higher overhead than most other techniques, as it requires training a secondary model simultaneously. Finally, ensembles have the highest training overhead as they require training five models.

V. Conclusions

Faults in training datasets for machine learning (ML) applications are often unavoidable. Such faults affect inference outcomes, which can be significant in safety-critical settings. Therefore, it is important to mitigate training data faults.

We explored the effectiveness of different techniques that have been proposed to deal with faults in training data, under different datasets, ML models, and fault rates and types. We find that using an *ensemble of multiple ML models* is the most effective approach to dealing with training data faults. Ensembles, however, require significant extra resources (for both training and inference). *Label smoothing*, which comes next in terms of effectiveness, is a practical alternative under resource constraints as it incurs much lower overheads. As future work, we will expand our evaluation to other data types.

All our experimental results and code available at:
https://github.com/DependableSystemsLab/TDFM-Techniques

ACKNOWLEDGMENTS

This work was funded in part by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC). We thank Phil Koopman for helpful discussions, and Saurabh Bagchi and the DSN'22 reviewers for their insightful comments.

REFERENCES

[1] J. G. Richens, C. M. Lee, and S. Johri, "Improving the accuracy of medical diagnosis with causal machine learning," *Nature Communications*, vol. 11, no. 1, p. 3923, 2020.

[2] S. S. Banerjee, S. Jha, J. Cyriac, Z. T. Kalbarczyk, and R. K. Iyer, "Hands Off the Wheel in Autonomous Vehicles?: A Systems Perspective on over a Million Miles of Field Data," in *Proc. of DSN'18*, 2018.

[3] T. Ozturk, M. Talo, E. A. Yildirim, U. B. Baloglu, O. Yildirim, and U. Rajendra Acharya, "Automated detection of COVID-19 cases using deep neural networks with X-ray images," *Computers in Biology and Medicine*, vol. 121, p. 103792, 2020.

[4] A. J. DeGrave, J. D. Janizek, and S.-I. Lee, "AI for radiographic COVID-19 detection selects shortcuts over signal," *Nature Machine Intel*, 2021.

[5] A. Laghi, "Cautions about radiologic diagnosis of COVID-19 infection driven by artificial intelligence," *The Lancet Digital Health*, vol. 2, 2020.

[6] A. Khetan, Z. C. Lipton, and A. Anandkumar, "Learning From Noisy Singly-labeled Data," 2018.

[7] Canadian Association of Radiologists (CAR) Artificial Intelligence Working Group, "Canadian Association of Radiologists White Paper on Ethical and Legal Issues Related to Artificial Intelligence in Radiology," *Canadian Association of Radiologists Journal*, vol. 70, no. 2, pp. 107–118, 2019.

[8] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, "ChestX-Ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

[9] S. Tang, A. Ghorbani, R. Yamashita, S. Rehman, J. A. Dunnmon, J. Zou, and D. L. Rubin, "Data Valuation for Medical Imaging Using Shapley Value: Application on A Large-scale Chest X-ray Dataset," *Scientific Reports*, vol. 11, no. 1, 2021.

[10] "A popular self-driving car dataset is missing labels for hundreds of pedestrians," https://blog.roboflow.com/self-driving-car-dataset-missing-pedestrians/, accessed: 2021-12-01.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *Proc. of CVPR'09*, 2009.

[12] C. G. Northcutt, A. Athalye, and J. Mueller, "Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks," 2021.

[13] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, "Learning from Noisy Labels with Deep Neural Networks: A Survey," 2021.

[14] N. e. a. Peri, *Deep k-NN Defense Against Clean-Label Data Poisoning Attacks*, 2020.

[15] A. Paudice, L. Muñoz-González, and E. Lupu, "Label Sanitization against Label Flipping Poisoning Attacks," 2018.

[16] J. Lienen and E. Hüllermeier, "From Label Smoothing to Label Relaxation," in *Proc. of AAAI'21*, 2021.

[17] G. Zheng, A. H. Awadallah, and S. Dumais, "Meta Label Correction for Noisy Label Learning," in *Proc. of AAAI'21*, 2021.

[18] X. Ma, H. Huang, Y. Wang, S. Romano, S. Erfani, and J. Bailey, "Normalized Loss Functions for Deep Learning with Noisy Labels," in *Proc. of ICML'20*, 2020.

[19] L. Zhang, J. Song, A. Gao, J. Chen, C. Bao, and K. Ma, "Be Your Own Teacher: Improve the Performance of Convolutional Neural Networks via Self Distillation," 2019.

[20] M. A. Hedeya, A. H. Eid, and R. F. Abdel-Kader, "A Super-Learner Ensemble of Deep Networks for Vehicle-Type Classification," *IEEE Access*, vol. 8, pp. 98 266–98 280, 2020.

[21] A. Chan, N. Narayananan, A. Gujarati, K. Pattabiraman, and S. Gopalakrishnan, "Understanding the Resilience of Neural Network Ensembles against Faulty Training Data," in *Proc. of QRS'21*, 2021.

[22] B. L. Sturm, "The GTZAN dataset: Its contents, its faults, their effects on evaluation, and its future use," *CoRR*, 2013. [Online]. Available: http://arxiv.org/abs/1306.1461

[23] N. Sambasivan, S. Kapania, H. Highfill, D. Akrong, P. K. Paritosh, and L. M. Aroyo, ""Everyone wants to do the model work, not the data work": Data Cascades in High-Stakes AI," in *Proc. of CHI'21*, 2021.

[24] D. Kermany, K. Zhang, and M. Goldbaum, "Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification," https://data.mendeley.com/datasets/rscbjbr9sj/2, 2018.

[25] T. Rädsch, S. Eckhardt, F. Leiser, K. Pandl, S. Thiebes, and A. Sunyaev, "What Your Radiologist Might be Missing: Using Machine Learning to Identify Mislabeled Instances of X-ray Images," in *Proc. of HICSS'21*, 2021.

[26] M. B. et al., "Active label cleaning: Improving dataset quality under resource constraints," 2021.

[27] M. Lukasik, S. Bhojanapalli, A. K. Menon, and S. Kumar, "Does label smoothing mitigate label noise?" 2020.

[28] C.-B. Zhang, P.-T. Jiang, Q. Hou, Y. Wei, Q. Han, Z. Li, and M.-M. Cheng, "Delving Deep Into Label Smoothing," *IEEE Transactions on Image Processing*, p. 5984–5996, 2021.

[29] X. Wang, Y. Hua, E. Kodirov, D. A. Clifton, and N. M. Robertson, "ProSelfLC: Progressive Self Label Correction for Training Robust Deep Neural Networks," 2021.

[30] J. Han, P. Luo, and X. Wang, "Deep Self-Learning From Noisy Labels," 2019.

[31] N. Charoenphakdee, J. Lee, and M. Sugiyama, "On Symmetric Losses for Learning from Corrupted Labels," 2019.

[32] Z. Zhang and M. R. Sabuncu, "Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels," 2018.

[33] Z. Li, X. Wang, H. Yang, D. Hu, N. M. Robertson, D. A. Clifton, and C. Meinel, "Not All Knowledge Is Created Equal," 2021.

[34] H. Shah, A. Khare, N. Shah, and K. Siddiqui, "KD-Lib: A PyTorch library for Knowledge Distillation, Pruning and Quantization," 2020.

[35] J. Lee and S.-Y. Chung, "Robust Training with Ensemble Consensus," 2020.

[36] D. T. Nguyen, C. K. Mummadi, T. P. N. Ngo, T. H. P. Nguyen, L. Beggel, and T. Brox, "SELF: Learning to Filter Noisy Labels with Self-Ensembling," 2019.

[37] D. Karimi, H. Dou, S. K. Warfield, and A. Gholipour, "Deep learning with noisy labels: Exploring techniques and remedies in medical image analysis," 2020.

[38] B. Frénay and M. Verleysen, "Classification in the Presence of Label Noise: A Survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 845–869, 2013.

[39] F. R. Cordeiro and G. Carneiro, "A Survey on Deep Learning with Noisy Labels: How to train your model when you cannot trust on the annotations?" 2020.

[40] M. Smieja, Łukasz Struski, J. Tabor, B. Zieliński, and P. Spurek, "Processing of missing data by neural networks," 2019.

[41] N. B. Ipsen, P.-A. Mattei, and J. Frellsen, "How to deal with missing data in supervised deep learning?" in *Artemiss - ICML*

Workshop on the Art of Learning with Missing Values, 2020.

[42] L. Bossard, M. Guillaumin, and L. Van Gool, “Food-101 – Mining Discriminative Components with Random Forests,” in *European Conference on Computer Vision*, 2014.

[43] M. Raghu, C. Zhang, J. Kleinberg, and S. Bengio, “Transfusion: Understanding Transfer Learning for Medical Imaging,” 2019.

[44] M. Abadi *et al.*, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015. [Online]. Available: https://www.tensorflow.org/

[45] R. Müller, S. Kornblith, and G. Hinton, “When does label smoothing help?” 2020.

[46] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2013.

[47] A. Ghosh, H. Kumar, and P. S. Sastry, “Robust Loss Functions under Label Noise for Deep Neural Networks,” 2017.

[48] G. Hinton, O. Vinyals, and J. Dean, “Distilling the Knowledge in a Neural Network,” 2015.

[49] A. Krizhevsky, “Learning multiple layers of features from tiny images,” Tech. Rep., 2009.

[50] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, “Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition,” *Neural Networks*, no. 0, pp. – , 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608012000457

[51] N. Narayanan and K. Pattabiraman, “TF-DM: Tool for Studying ML Model Resilience to Data Faults,” in *Proc. of DeepTest’21*, 2021.