Robust Federated Learning via Collaborative Machine Teaching

Yufei Han
Nortonlifelock Research Group
Campus SohpiaTech
Sophia Antipolis, 06410, France
yufei_han@symantec.com

Xiangliang Zhang
Machine Intelligence and Knowledge Engineering Laboratory
King Abudullah University of Science and Technology
Thuwal,23955,Kingdom of Saudi Arabia
xiangliang.zhang@kaust.edu.sa

Abstract

For federated learning systems deployed in the wild, data flaws hosted on local agents are widely witnessed. On one hand, given a large amount (e.g. over 60%) of training data are corrupted by systematic sensor noise and environmental perturbations, the performances of federated model training can be degraded significantly. On the other hand, it is prohibitively expensive for either clients or service providers to set up manual sanitary checks to verify the quality of data instances. In our study, we echo this challenge by proposing a collaborative and privacy-preserving machine teaching method. Specifically, we use a few trusted instances provided by teachers as benign examples in the teaching process. Our collaborative teaching approach seeks jointly the optimal tuning on the distributed training set, such that the model learned from the tuned training set predicts labels of the trusted items correctly. The proposed method couples the process of teaching and learning and thus produces directly a robust prediction model despite the extremely pervasive systematic data corruption. The experimental study on real benchmark data sets demonstrates the validity of our method.

Introduction

Federated learning is well known for its vulnerability to abnormal behavior of individual computing agents (Yang et al. 2019). The anomaly is caused by unreliable computing devices, communication jams or flaws existing in the local training data sets. Our study focuses on the impact of **pervasive training data flaws**, where a significantly large fraction of the local data sets (e.g. over 60%) are contaminated by systematic noise corruption, such as **non-iid feature corruption** and **asymmetric label flipping noise**. In real-world distributed data analytic applications, such training data flaws are widely witnessed due to malfunction of data collectors, crowd-sourcing and intentional adversarial attacks from hacked / untrustworthy participating devices. On one hand, the severe noise corruption can deeply deteriorate learning performances. On the other hand, manual inspection and verification of each data instance is usually expensive thus unrealistic for both clients and service providers to conduct pre-training data quality check. Deployment of the existing federated training method with the pervasive data flaws carries a threat to the accountability of data analytic service.

Recent robustification efforts improve resilience to Byzantine failure (Chen et al. 2018a; Chen, Su, and Xu 2017; Yin et al. 2018; Xie, Koyejo, and Gupta 2018b; 2018a; Alistarh, Allen-Zhu, and Li 2018; Su and Xu 2019; Blanchard et al. 2017; Ghosh et al. 2019; Li et al. 2018). They assume that the majority of local agents behave normally and consider the Byzantine machines as statistical outliers. Robust estimation is then applied to suppress the negative impact of Byzantine machines in model/gradient averaging. Nevertheless, given the pervasive systematic data flaws, the profiles of the distributed training data set can be drastically twisted. Furthermore, the robust estimation is applied based on the assumption over data distribution, such as long-tailed distribution. However, due to the privacy regulation of federated learning, prior knowledge about the noise distribution on local agents is barely available. The robust estimation based approaches thus become less effective in this challenging scenario.

We propose to mitigate the adverse impact of the pervasive systematic data corruption in federated training via **C**ollaborative **M**achine **T**eaching, thus named ***CoMT***. At the core of *CoMT*, the local *agents* act as distributed *teachers*, while the central *parameter server* is the *learner* cotaught by the distributed *teachers*. Besides the corrupted training data, each teacher also hosts a few trusted data instances, e.g. verified by domain experts, to guide the teaching process. The teachers are organized to jointly tune the corrupted training data set, such that the model learnt with the tuned training set by the learner predicts consistent targets as the trusted instances. More specifically, we adopt two types of training data tuning operations: **crafting the training instances** and **subset-selecting the training set**. The set of the trusted data instances are small in size, e.g. $1/100$ of the training data. They are thus insufficient for learning by themselves. The merits of the trusted instance-directed training set tuning are three-fold. First, the training set tuning requires no assumption on noise distribution. Compared to the robust-statistics based methods, the assumption-free method reduces the risk of miss-estimating noise corruption due to lack of prior knowledge of noise distribution. Second, the trusted instances acts as weak supervision. The teaching activity with a learning-performance directed objective en-

courages the training data tuning efforts to help recover the underlying feature-label relation. Furthermore, our method couples federated teaching and learning in the joint optimization process, which produces the learned model as output directly. Finally, we define the objective function of the teaching collaboration based on the dual form of the learning paradigm of the learner. It facilitates designing a privacy-preserving and efficient optimization subroutine with block coordinate descent. Training instances of one teacher never leave the hosting machine and are not accessible by the other teachers during the optimization process.

Related Work

Byzantine-robust federated learning has received increasing attention in recent years. Most existing algorithms assume that data are independent and identically distributed (i.i.d.). Local model updates computed by normal machines are around the true gradient, while those sent from the Byzantine machines or malicious adversary agents (Bhahpji et al. 2018; Chen et al. 2018a) are outliers. Robust statistics, e.g. median-based and weighted-based aggregation, are applied on the central parameter server to achieve noise-tolerant learning. The works along this direction include geometric median (Chen, Su, and Xu 2017), marginal trimmed mean (Yin et al. 2018; Xie, Koyejo, and Gupta 2018b), dimensional median (Xie, Koyejo, and Gupta 2018a; Alistarh, Allen-Zhu, and Li 2018) and iterative filtering (Su and Xu 2019). A more sophisticated algorithm named as Krum (Blanchard et al. 2017) selects a gradient with minimal summation of Euclidean distances from a given number of nearest gradients. Recent study (Ghosh et al. 2019; Li et al. 2018) study Byzantine attack against federated learning systems with heterogeneous data distributions. (Ghosh et al. 2019) proposes to perform clustering of local agents based on the expected risk minimizers derived on each agent. Training data within each cluster have more homogeneous distributional characteristics. The robust aggregation approaches, e.g. geometric median, is then applied in each cluster. (Li et al. 2018) maintains stochastic gradient calculation on both local agents and the master. It pursues to minimize the sum of L1-norm distance between local estimates of model parameters and that derived on the master, which limits the influences of Byzantine machines.

Machine teaching (Goldman and Kearns 1995) proposes an inverse problem to machine learning: how does a teacher construct a training data set to help a learner achieve the teaching goal? Most of the pioneering works (Liu, Zhu, and Ohannessian 2016; Zhu 2013) focus on studying a key quantity called the teaching dimension, i.e., the size of the minimal training set that is guaranteed to teach a target model to the learner. Later works consider other variants of teaching setting, e.g.,in (Zilles et al. 2011; Balbach 2008), the learner and the teacher are allowed to cooperate in order to achieve better teaching performance. More theoretical studies about machine teaching can be found in (Doliwa et al. 2014; Chen et al. 2018b; Haug, Tschiatschek, and Singla 2018; Liu et al. 2017b; 2017a). Our work is aligned with *Super Teaching* (Ma et al. 2018) and *Learning to Teach* (Fan et al. 2018) on selecting training instances. Another closely related work is *Debugging Using Trusted Items* (DUTI) (Zhang, Zhu, and Wright 2018). *DUTI* defines training data flaws as noise corruption over the labels/targets of the training data. Teaching task of *DUTI* is to inject the minimal changes to the training targets, such that the trained model predicts consistent output with the trusted instances. The magnitudes of the the injected crafting efforts are then used to flag potential corrupted targets. Main limits of the previous machine teaching methods lie in three folds. First, they focus on theoretical teachability study with toy scenarios, e.g. how to make the learner learn the specified parameter values with tuned training data. They are thus not practically useful, as the parameter values of the target model are barely known. In contrast, the proposed *CoMT* produces directly an applicable prediction model. Second, the previous works involve only one teacher. Few effort has been engaged to study how to organize collaboration between distributed teachers and protect data privacy of each teacher simultaneously. The proposed *CoMT* achieves both goals via a block coordinate descent based optimization method. Finally, previous machine teaching methods suffer from intense computation for solving a bi-level stackelberg game (Brückner and Scheffer 2011) with non-linear programming or combinatorial optimization techniques. Benefited from the parallel block coordinate descent method, we can organize the teaching collaboration efficiently

Collaborative Machine Teaching with Trusted Instances

Assuming that there are K teachers, each of them hosts a buggy training set $\{(X_i^k, Y_i^k)\}_{i=1:n_k}$. X^k and Y^k denote the features and labels of one local training set. In addition, we assume that each agent provides a small portion of trusted data instances $\{(\tilde{X}_i^k, \tilde{Y}_i^k)\}_{i=1:m_k}$, where $m_k \ll n_k$. The proposed collaborative teaching is defined as a bi-level optimization problem:

$$
\begin{aligned}
&\min_{X', Y', b^k : k\in[K]} \sum_{k=1}^{K}\sum_{i=1}^{n_k} b_i^k \{d_x(X_i'^k, X_i^k) + d_y(Y_i'^k, Y_i^k)\} + \lambda_b |b| \\
&s.t.\ \ \hat{\theta}^* = \arg\min_{\hat{\theta}} \sum_{k=1}^{K}\sum_{i=1}^{n_k} b_i^k \ell(\hat{\theta}, X_i^{'k}, Y_i^{'k}) + \lambda\Omega(\hat{\theta}), \\
&b_i^k = \{0,1\},\ f_{\hat{\theta}^*}(\tilde{X}_i^k) = \tilde{Y}_i^k,\ \ f_{\hat{\theta}^*}(X_i'^k) = Y_i'^k
\end{aligned}
\tag{1}
$$

where d_x measures Euclidean distance between the changed feature $X_i'^k$ and X_i^k. d_y denotes Euclidean distance (for regression) or hamming distance (for classification) between $Y_i'^k$ and Y_i^k. $|b|$ denotes L1-norm of b. $\sum_{k,i} \ell(\hat{\theta}, X_i'^k, Y_i^k) + \lambda\Omega(\hat{\theta})$ denotes the regularized learning paradigm $\mathcal{A}$ of the learner to train the model f parameterized by $\hat{\theta}$. Minimizing Eq.1 simultaneously selects an informative subset of training data and inject the minimum changes into the the selected data instances. The tuned training subset is used to conduct federated training with the learning paradigm $\mathcal{A}$. The learnt model f should predict consistent labels on both trusted instances and the tuned training data.

As b_i^k is binary, directly solving Eq.1 reduces to a mixed-integer non-linear constraint problem (MINLP). In the worst case, the popular heuristic solver, such as Branch-and-Bound method, has an exponential time complexity thus becomes prohibitively expensive given large-scale training data. Solving a MINLP problem with distributed players is even more difficult due to the privacy concern. It usually needs a central processor to access the data of local machines (Karabulut 2017), which violates the privacy regulation. Besides, frequent synchronization between the central process and the end-devices can cause severe latency given a low-communication environment. We attack this issue by reformulating the bi-level objective of *CoMT* with the dual form of the learning paradigm $\mathcal{A}$.

Dual form of the collaborative teaching

The dual objective of the learning paradigm $\mathcal{A}$ of the learner gives as:

$$\alpha^* = \arg\min_{\alpha} \sum_{k=1}^{K}\sum_{i=1}^{n_k} \ell^*(-\alpha_i^k) + \frac{\lambda}{2}\|Z\alpha\|^2 \tag{2}$$

where ℓ^* is the Fenchel dual of the loss function ℓ. Let $n = \sum_{k=1}^{K} n_k$ denote the total number of training instances owned by the teachers. $Z \in R^{d*n}$ denotes aggregated data matrix with each column corresponding to a data instance. The duality comes with the mapping from dual to primal variable: $\omega(\alpha) = Z\alpha$ according to the KKT optimality condition. Each α_i^k as the dual variable corresponding to the ith data instance hosted by teacher k. If α_i^k diminishes to zero, the corresponding data instance Z_i^k consequently has no contribution to the dual objective in Eq. (2). Thus, only the data instances with non-zero α_i^k dominates the training process. Motivated by this observation, we can reformulate Eq.1, the objective of the proposed collaborative teaching method following Eq.3:

$$\begin{aligned}\alpha, Z' = &\arg\min_{\alpha_i^k, k\in[K], Z'} \frac{1}{n}\sum_{k=1}^{K}\sum_{i=1}^{n_k} \ell^*(-\alpha_i^k, Z') + \frac{\lambda}{2}\|Z'\alpha\|^2 \\ &+ \lambda_t \frac{1}{m}\sum_{k=1}^{K}\sum_{i=1}^{m_k} \ell(\tilde{X}_i^k, \tilde{Y}_i^k, Z'\alpha) + \lambda_Z\|Z' - Z\|^2 + \lambda_\alpha \sum_{k=1}^{K}|\alpha^k|\end{aligned} \tag{3}$$

where λ_t balances the impact of the trusted data instances in the joint optimization process. Larger λ_t sets more strict constraints over the consistency between the learned model and the trusted data. λ_α is the regularization weight of the adaptive l_1-norm penalization enforcing sparsity of α to perform the subset selection. λ_Z penalizes the teaching efforts applied to the corrupted training data. An appropriately chosen λ_Z prevents too much artefacts introduced to the tuned training instances, while still enables the tuning flexibility to deliver efficient teaching. The teaching collaboration defined by Eq. (3) is convex according to the property of Legendre-Fenchel transform. Thus solving Eq. (3) with gradient descent guarantees fast convergence. As we enforce a sparsity regularization over α, the magnitudes of the non-zero entries of α can be used to select training instances to calculate the model parameters. We next demonstrate how to apply the proposed *CoMT* method to two prevalent learners, L2-regularized Logistic Regression (LR) and Ridge Regression (RR).

CoMT for Ridge Regression and Logistic Regression

We instantiate *CoMT* with Ridge Regression as the learner by inserting the dual form of Ridge Regression into Eq.3, as given as follows:

$$\begin{aligned}\alpha^{*,k}, \beta_{x,y}^{*,k} = &\arg\min_{\alpha^k, \beta_x^k, \beta_y^k} \frac{1}{2\lambda_w}\sum_{k=1}^{K}\|(X^k + \beta_x^k)\alpha^k\|^2 + \frac{1}{2}\sum_{k=1}^{K}\|\alpha^k\|^2 \\ &- \sum_{k=1}^{K}\alpha^{k,T}(Y^k + \beta_y^k) + \lambda_t\sum_{k=1}^{K}\|\tilde{X}^k\frac{1}{\lambda_w}(\sum_{k=1}^{K}(X^k + \beta^k)\alpha^k) - \tilde{Y}^k\|^2 \\ &+ \lambda_\alpha\sum_{k=1}^{K}|\alpha^k| + \lambda_Z\sum_{k=1}^{K}(\|\beta_x^k\|^2 + \|\beta_y^k\|^2)\end{aligned} \tag{4}$$

where λ_w is the regularization parameter over the regression coefficients w in ridge regression. β_x^k and β_y^k denote the teaching crafting on the feature and target of each training data instance. The magnitude of each element in α^k measures proportionally the contribution of each training data instance hosted by one teacher in learning. The first three terms of Eq.4 enforce the consistency between the learned regression parameter w and the changed training data instances $(X^k + \beta^k)$ and Y^k. They are derived as the dual definition of Ridge Regression. The forth term enforces the consistency of the learnt model with the trusted instances. Similarly, *CoMT* for L2-regularized Logistic Regression in can be defined as in Eq.(5):

$$\begin{aligned}\alpha^*, \beta^* = &\arg\min_{\alpha^k, \beta^k} \sum_{k=1}^{K}\sum_{i=1}^{n_k}(\alpha_i^k\log(\alpha_i^k) + (1-\alpha_i^k)\log(1-\alpha_i^k)) \\ &+ \lambda_w\|w\|^2 + \lambda_t\sum_{k=1}^{K}\sum_{i=1}^{m_k}\log(1 + exp(-\tilde{Y}_i^k w\tilde{X}_i^k)) \\ &+ \lambda_\alpha\sum_{k=1}^{K}|\alpha^k| + \lambda_Z\sum_{k=1}^{K}\|\beta^k\|^2 \;\; s.t. \;\; 0 < \alpha_i^k < 1\end{aligned} \tag{5}$$

where $w = \frac{1}{\lambda_w}\sum_{k=1}^{K}\sum_{i=1}^{n_k}\alpha_i^k(Z_i^k + \beta_i^k)$. Benefited from the dual form of LR, we apply the data crafting based teaching directly on the classification margin of training instances in Eq.5. It avoids the combinatorial optimization of tuning the label Y'. Instead, it solves a much easier L1-regularized continuous programming problem. Note that *CoMT* can be easily adapted to the case with non-linear kernel learners by introducing random Fourier features (Rahimi and Recht 2007).

CoMT optimization

We design a privacy-preserving optimization subroutine to solve the teaching collaboration defined by Eq.4 and Eq.5. Our method is inspired by dual coordinate descent (Smith

et al. 2018) and Alternating Direction Method of Multiplier (ADMM). In each round of the descent process, we conduct minimization with w.r.t. α_i^k and β_i^k belonging to the k-th local agent, while fixing all the other α^k and β^k. We take *CoMT* for RR as an example. Similar process can be adapted to LR.

We first reformulate Eq.4 with scaled ADMM, as shown in Eq.6:

$$\begin{aligned}
&\alpha^{k,*}, \beta_{x,y}^{k,*} = \arg\min_{\alpha^k, \beta_x^k, \beta_y^k} \frac{1}{2\lambda_w} \sum_{k=1}^{K} \|(X^k + \beta_x^k)\alpha^k\|^2 + \frac{1}{2} \sum_{k=1}^{K} \|\alpha^k\|^2 \\
&- \sum_{k=1}^{K} \alpha^{k,T}(Y^k + \beta_y^k) + \lambda_\alpha \sum_{k=1}^{K} |\alpha^k| + \lambda_Z \sum_{k=1}^{K} (\|\beta_x^k\|^2 + \|\beta_y^k\|^2) \\
&+ \frac{\rho}{2} \|\widetilde{\theta} - w + u\|^2 \\
&\widetilde{\theta}^* = \arg\min_{\widetilde{\theta}} \lambda_t \sum_{k=1}^{K} \|\tilde{X}^k \widetilde{\theta} - \tilde{Y}^k\|^2 + \frac{\rho}{2} \|\widetilde{\theta} - w + u\|^2 \\
&u = u + \widetilde{\theta} - w
\end{aligned} \tag{6}$$

where $w = \frac{1}{\lambda_w}(\sum_{k=1}^{K}(X^k + \beta_x^k)\alpha^k)$. $\rho > 0$ is the augmented Lagrangian parameter and u is the scaled dual variable of ADMM. The pseudo codes of the optimization procedure are given in Algorithm 1.

$\alpha^{t,k}$ and $\beta^{t,k}$ denote the value of the disjoint block $\{\alpha_i^k, \beta_i^k\}_{i=1,..,n_k}$ estimated at the t-th iteration. They correspond to the data instances hosted by the k-th teacher. In each round of iteration, we update the dual variables $\alpha^{t,k}$ and $\beta^{t,k}$ of each teacher in parallel. The incremental updates $\Delta\alpha^k$ and $\Delta\beta^k$ are estimated based on the value of $\alpha^{t-1,k}$ and $\beta^{t-1,k}$ and minimize the linear local approximation to Eq. (6). The updated $\alpha^{t,k}$ and $\beta^{t,k}$ of each teacher are then used to aggregate τ^t. After that, we solve a quadratic programming problem for each teacher to compute the local update $\Delta\widetilde{\theta}^k$ and take the average of them to update $\widetilde{\theta}$.

It is easy to find that: i) training data of any local teacher never leave the hosting machine in the collaboration stage. Furthermore, updating $\widetilde{\theta}$ only needs to aggregate local updates on the learner to derive τ^t and broadcast it back to the teachers. The learner can barely reverse-engineer the statistical profiles of the training data of local teachers with only τ^t and $\widetilde{\theta}^t$. It prevents the risk of unveiling private data of one local teacher to the learner or the other teachers. ii) Information sharing between local teachers is conducted by updating the global variable $\widetilde{\theta}$ and τ in Algorithm.1. Communication for teaching and learning collaboration is thus efficient, with the cost of $O(Kd + nd)$ in each round of iteration. Moreover, according to (Jaggi et al. 2014), updating α^k of local teachers can be triggered with asynchronous parallelism, which allows to organize efficient collaboration of teaching and learning with tight communication budget. Note that most entries of α are tuned to be close to zeros. We identify the data instances corresponding to the entries of α with the largest non-zero magnitudes. Only the selected data instances are used to calculate the model parameter. Once α^k and β^k are derived, we can obtain the model parameter as $\frac{1}{\lambda_w}\sum_{k=1}^{K}\sum_{\tilde{i}\in S_k}\alpha_{\tilde{i}}^k(X_{\tilde{i}}^k + \beta_{\tilde{i}}^k)$, with S_k as the set of selected training instances of the teacher k.

Data: Buggy training data $\{X_i^k, Y_i^k\}_{k=1,...,K, i=1,...,n_k}$ and the trusted data $\{\tilde{X}_i^k, \tilde{Y}_i^k\}_{k=1,...,K, i=1,...,m_k}$ hosted by K teachers

Input: $T \geq 1$ as the maximum iteration steps, scaling parameter $1 \leq \gamma_k \leq K$, by default $\gamma_k = 1$. ρ = 1e2 as the augmented Lagrangian multiplier

Output: $\alpha_i^{T,k}, \beta_i^{T,k}, k = 1, 2, ..., K, i = 1, 2, ..., n_k$

Initialize: Set $\alpha_i^{0,k} = 0$ and $\beta_i^{0,k} = 0$ for all K machines. $\widetilde{\theta}^{(0)} = 0$ and $u^0 = 0$

for $t = 1$ **to** T **do**

  for *all teachers* $k = 1, 2, 3, ..., K$ *in parallel* **do**

    $\Delta\alpha^{k,*}, \Delta\beta^{k,*} = \arg\min_{\Delta\alpha^k, \Delta\beta^k} \frac{1}{2\lambda_w}\|(X^k + \beta^{t-1,k} + \Delta\beta^k)(\alpha^{t-1,k} + \Delta\alpha^k)\|^2 + \frac{1}{2}\|\alpha^{t-1,k} + \Delta\alpha^k\|^2 - (\alpha^{t-1,k} + \Delta\alpha^k)^T Y^k + \lambda_\alpha|\alpha^{t-1,k} + \Delta\alpha^k| + \lambda_Z\|\beta^{t-1,k} + \Delta\beta^k\|^2 + \frac{\rho}{2}\|\widetilde{\theta}^{t-1} - \frac{1}{\lambda_w}(X^k + \beta^{t-1,k} + \Delta\beta^k)(\alpha^{t-1,k} + \Delta\alpha^k) + u^{t-1}\|^2$

    $\alpha^{t,k} = \alpha^{t-1,k} + \frac{\gamma_k}{K}\Delta\alpha^k$

    $\beta^{t,k} = \beta^{t-1,k} + \frac{\gamma_k}{K}\Delta\beta^k$

  end

  Reduce on the central parameter server

  $\tau^t = \frac{1}{\lambda_w}\sum_{k=1}^{K}\sum_{i=1}^{n_k}\alpha_i^{t,k}(X_i^k + \beta_i^{t,k})$

  Broadcast τ^t *to all* K *teachers*

  for *all* K *teachers* $k = 1, 2, 3, ...K$ *in parallel* **do**

    $\Delta\widetilde{\theta}^k = \arg\min_{\Delta\widetilde{\theta}} \lambda_t\|\tilde{X}^k(\widetilde{\theta} + \Delta\widetilde{\theta}^k) - \tilde{Y}^k\|^2 + \frac{\rho}{2}\|(\widetilde{\theta} + \Delta\widetilde{\theta}^k) - \tau^t + u^{t-1}\|^2$

  end

  Reduce on the central learner

  $\widetilde{\theta}^t = \widetilde{\theta}^{t-1} + \frac{1}{K}\sum_{k=1}^{K}\Delta\widetilde{\theta}^k$

  Update on the central learner

  $u^t = u^{t-1} + \widetilde{\theta}^t - \tau^t$

  Broadcast $\widetilde{\theta}^t$ *and* u^t *to all* K *teachers*

end

Algorithm 1: Block-Coordinate Descent for CoMT

Empirical Study

We involve the following baseline approaches in the study:

- **TI-only** and **TI-Noise** uses only the trusted instances and both trusted and corrupted training data of by each teacher to conduct model training. We include them to confirm that the proposed machine teaching method can extract useful information from buggy data for training
- **DUTI** (Zhang, Zhu, and Wright 2018) is the most relevant with our work. In this study, each individual teacher runs *DUTI* independently with his own local data set. After that, the tuned local training data sets of all the teachers are used for federated model training. As DUTI is not designed for distributed computing, we use this setting to be

aligned with the federated learning scenario, where neither the central server nor any of the local teacher is able to access the whole data set.

- **REWLS** (Gervini and Yohai 2002),**RloR** (Feng et al. 2014) and **rLR** (Bootkrajang and Kaban 2012) are robustified ridge regression and logistic regression. *REWLS* is a weighted least square estimator with the weights adaptively calculated from an initial robust estimator. *RloR* and *rLR* are robust against outliers with feature and label noise respectively. We use them as the baselines with robust loss functions. We apply them on each local teacher independently to derive local parameter estimate. These local estimates are aggregated to derive model parameters.
- **GeoMed** (Chen, Su, and Xu 2017) and **RSA**(Li et al. 2018) are robust stochastic gradient based optimization methods. Both methods gain significant robustness improvement to Byzantine machines according to (Li et al. 2018). They compute robust estimation of model update using noise-biased stochastic gradients calculated by each local machine. In the following experiments, we use them with the standard learning objectives of ridge regression and L2-regularized logistic regression.

DUTI, *REWLS*, *RloR*, *rLR*, *GeoMed* and *RSA* are trained with the trusted instances and the corrupted data. For *DUTI*, we choose a sequence of sparsity parameters to generate a series of flagged buggy data sets, until the size of the union of these sets exceeds the examination budget. The parameters of *REWLS*, *RloR* and *rLR* are chosen according to the ratio of the buggy instances in the training data. The hyperparameters of *GeoMed* and *RSA* are chosen as suggested in (Chen, Su, and Xu 2017) and (Li et al. 2018). 4 large-scale real-world data sets with different application contexts are used to benchmark the involved algorithms (summarized in Table.1)(Chang and Lin 2011). Without loss of generality, we fix the number of teachers K to 5 and assume that each teacher hosts the same number of training instances. For each real-world data set, we first randomly extract 60% of the whole data set as the training data. These training data are corrupted to generate buggy training data. Similarly, for each data set of *CPUSMALL*, *ABALONE* and *IJCNN*, we choose $\eta = 1.0\%$ and $\eta = 5.0\%$ of the whole data set as the trusted instances. For *SUSY*, training with trusted instances more than 1% of the data set can achieve similar AUC scores as that derived with the ground truth of the training data. Therefore, we set η as 0.1% and 0.5%. These buggy training data and the trusted instances are assigned to the K teachers uniformly. It is worth noting that we uniformly assign data to different teachers only to avoid unnecessarily sophisticated experimental setting. In addition, according to Eq.4 and Eq.5, it is not obliged to have every teacher provide the trusted instances. Only the number of the trusted instances matters in seeking the optimal teaching efforts. To tune the parameter λ_t, λ_α and λ_Z, we adopt 10% of the data as the validation set. The rest of the data instances are used to evaluate the performances of the learned model. For regression test, we follow Eq.7 to add random noise to the feature vector and regression target of each training instance:

$$X_i = \hat{X}_i + \vartheta_x,\ Y_i = \hat{Y}_i + \vartheta_y \tag{7}$$

Table 1: Summary of 4 real-world benchmark datasets.

Dataset	No. of Instances	No. of Features
IJCNN	49,990	22
SUSY	50,000	18
CPUSMALL	8,192	12
ABALONE	4,177	8

where $\hat{X}$ and $\hat{Y}$ are the original feature and target of the training instances. ϑ_x and ϑ_y are the injected noise following normal distribution. We normalize the values of each feature dimension in $\hat{X}$ between -1 and $+1$. For each teacher, we sample uniformly the mean of ϑ_x between -5 and $+5$ and the mean of ϑ_y between $-2.5 * avg(|y_i|)$ and $2.5 * avg(|y_i|)$ respectively, where $avg(|y_i|)$ denotes the average absolute value of the target $\hat{Y}_i$. We vary the variances of ϑ_x and ϑ_y, noted as δ_x and δ_y, to change the strength of the injected noise. This setting is used to simulate **non-iid systematic noise**. In the classification test, we follow Eq.7 to add the systematic feature corruption to the training instances. For label flipping based corruption, we first randomly select $\varphi = 60\%$ and $\varphi = 80\%$ of training labels respectively. In the selected labels, we require that $2/3$ of them are positive labels and rest of them are negative ones. They are used to mimic **asymmetric label flipping noise**, where positive instances are more prone to miss-labeling. We repeat random sampling of training data and noise injection for 10 times. The mean and variance of R-squared and AUC scores of the trained model on the testing instances measure accuracy of regression and classification.

In addition, as *CoMT* conducts subset selection over the training data, we set a fixed threshold $th = 1e-4$ empirically over α^k of each teacher for all 4 benchmark databases. Only the training instances with their α_i^k larger than th are used to aggregate the model parameters following Algorithm.1. We thus record the fraction ϱ of the selected instances in the whole training data set in the results.

All the methods are implemented in Python 2.7 with *Numpy* and *Scipy* packages on a 5-core AWS EC2 public cloud server, with one core per teacher.

Results of the regression set

We consider separately *non-iid regression feature corruption* and *non-iid regression target corruption* in table.2 and table.3. Co-occurrence of feature and target corruption is left in our future study. The R-squared score derived by training with clean training data and trusted instances is shown as the ground truth. Generally, increasingly stronger noise corruption over all local machines causes degradation of performances of the baseline methods. The proposed *CoMT* method, in contrast, consistently achieves the best and stable regression accuracy. *TI-only* and *TI-Noise* doesn't help in learning, as the trusted instances by themselves are not informative enough to conquer the pervasive noise.

Though *DUTI* shares the same origin as our method, its teaching flexibility is limited from two perspectives: i) it constraints the amount of the injected data crafting aggressively via the sparse regularization over the crafting variable. Besides, it doesn't include subset selection as a complementary teaching operation. ii) Standalone use of *DUTI* on each

teacher involves no teaching collaboration, which makes it fail to reach a jointly optimal teaching maneuver. As a result, even though *DUTI* performs better than *TI+Noise*, its performance is still worse than that of *CoMT*.

REWLS, *GeoMed* and *RSA* make use of robust estimators to handle the training data flaws. They assume the noise-corrupted model update as outliers in a long-tailed distribution. Nevertheless, the pervasive noise change drastically the profiles of training data. The long-tailed distribution assumption does not hold in this situation. Consequently, these method doesn't help to robustify the learning results. On *CPUSMALL* and *ABALONE*, *CoMT* uses 60% to 70% of the training data instances to calculate the model parameter. We also measure on average the magnitudes of the data crafting efforts injected to the selected corrupted training instances. It is calculated as the averaged ratio $1/K \sum_{k=1}^{K}(\|\beta_x^k\|/\|X^k\| + \|\beta_y^k\|/\|Y^k\|)$. The averaged ratio is 22% on *CPUSMALL* and 21% on *ABALONE*. Note that our focus is not to recover the underlying ground truth of training data profiles. We aim at tuning the training data to achieve better prediction accuracy. Therefore, we allow relatively larger data crafting in teaching compared to that reported in *DUTI* (Zhang, Zhu, and Wright 2018), which provides more tuning flexibility. However, the magnitude of the injected data crafting is still a small fraction of that of a training data instance and controllable by appropriately setting λ_Z in Eq.3.

Results of the classification set

We separately study systematic corruption over the feature vectors and class labels, as shown in table.4 and table.5. The AUC scores derived by training with clean training data and trusted instances are given in the tables as the ground truth accuracy. In contrast to the baseline methods, *CoMT* uses no more than 70% of the training data to obtain distinctively higher classification accuracy. By comparison, simply incorporating trusted instances into training (in *TI+Noise*) or teaching with limited flexibility (in *DUTI*) fails to improve the learning performances.

The results also confirm the necessity of organizing collaboration among different teachers, compared to the standalone use of *DUTI* on each teacher. As discussed over the regression test results, robsut estimator based solutions, such as *RloR*, *rLR*, *GeoMed* and *RSA* can not mitigate the pervasively occuring systematic noise, e.g. over 60% of training instances are miss-labeled. As we can find in the tables, these approaches don't help to improve accuracy compared to *TI+Noise*.

We use $1/K \sum_{k=1}^{K} \|\beta^k\|/\|Z^k\|$ to evaluate the relative strength of the manipulation efforts on the classification margins. The averaged ratio is 22% on *IJCNN* and 24% on *SUSY*, which is similar with those observed in the regression test. It demonstrates consistent applicability of *CoMT* in different learning scenarios.

Run-time cost test

Classic machine teaching methods suffer from the expensive cost of solving the bi-level teaching objective. To demonstrate that our work improves the applicability of machine teaching approaches on large-scale data, we measure the averaged run-time cost of the proposed *CoMT* method and *DUTI* on each data set, which includes the time cost for updating in parallel α^k and β^k of all teachers and communication between the teachers and the central learner.

Due to the space limit, we only choose the non-iid feature corruption scenario. For each data set, we fix $\eta = 0.10\%$ and choose the most contaminated setting of feature corruption. *CoMT* costs 35.00s, 45.51s, 264.32s and 359.75s respectively on *ABALONE*, *CPUSMALL*, *IJCNN* and *SUSY*. In contrast, the run-time cost of *DUTI* increases more drastically, which costs 45.54s and 227.40s on *ABALONE* and *CPUSMALL*, while more than 1000s on *IJCNN* and *SUSY*. *DUTI* requires to retrain the model after updating the data crafting variable in each iteration, whereas *CoMT* avoids this issue by introducing dual coordinate descent.

Limitations

The proposed **CoMT** approach inherits the idea of using trusted instances as seeds in the teaching process, which was originally proposed in (Zhang, Zhu, and Wright 2018). Likewise, it shares the similar theoretical limitations as those pointed out in (Zhang, Zhu, and Wright 2018). Intuitively, we ascribe the performance of CoMT to the teaching flexibility. CoMT allows both crafting and training subset selection based teaching. It increases the data manipulation space and thus offers more flexibility to search for consistent model parameters with the trusted instances. We believe that it is helpful for recovering the applicable model facing severe data flaws. Nevertheless, it is still an open issue to unveil theoretically how the trusted instances help to improve generalization capability of the learner. Such analysis is also helpful for evaluating quality of the trusted instances, in order to provide proper coverage of typical feature profiles. Besides, we select the trusted instance as i.i.d. samples. In contrast, trusted instances of typical feature profiles identified by field experts are usually highly correlated.

Conclusion and Future Work

In this paper, we explore how to produce a robust federated model training process despite of extremely pervasive systematic data corruption via collaborative teaching. In our work, conducting teaching efforts on the buggy training data and learning with the tuned data are coupled into a joint optimization problem. The learning-performance directed teaching helps recover the underlying true relation between the features and labels of training instances. Furthermore, the teaching collaboration is organized as a computationally efficient and privacy-preserving process. Information sharing between the collaborative teachers helps seek jointly optimal teaching efforts. The selected trusted instances should properly cover the typical feature profiles of domain data to guide the teaching activity. Our future work will focus on analyzing theoretically the role of trusted instances in improving the tolerance of the learner against data flaws. Furthermore, we plan to extend practical use of the proposed method over more complex learners, such as neural nets.

Table 2: Comparison of R-squared on *CPUSMALL* (**R-squared score with clean training data:**0.71)

Non-IID Regression Feature Corruption																
η	δ_x	***CoMT***			*TI-only*		*TI+Noise*		*DUTI*		*REWLS*		*GeoMed*		*RSA*	
		Avg	Var	ϱ	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var
1.0%	0.50	**0.69**	**2.73e-5**	**0.65**	0.62	7.43e-5	0.64	7.51e-5	0.64	4.51e-5	0.65	4.82e-5	0.64	5.75e-5	0.64	1.57e-5
	1.50	**0.68**	**2.50e-5**	**0.60**	0.62	7.43e-5	0.49	4.85e-5	0.48	6.70e-5	0.48	4.20e-5	0.49	7.49e-5	0.49	3.10e-5
	4.50	**0.69**	**2.34e-5**	**0.55**	0.62	7.43e-5	0.18	3.39e-5	0.29	4.66e-6	0.19	5.40e-6	0.18	3.05e-5	0.18	5.76e-5
5.0%	0.50	**0.67**	**3.55e-5**	**0.65**	0.67	1.47e-5	0.54	4.35e-5	0.55	2.73e-5	0.54	2.77e-5	0.54	4.61e-5	0.54	2.85e-5
	1.50	**0.71**	**2.51e-5**	**0.65**	0.67	1.47e-5	0.54	4.77e-5	0.57	3.20e-5	0.53	4.41e-5	0.54	4.59e-5	0.54	4.30e-5
	4.50	**0.71**	**1.92e-5**	**0.65**	0.67	1.47e-5	0.53	2.24e-5	0.57	3.68e-5	0.52	3.51e-5	0.52	4.01e-5	0.52	3.68e-5
Non-IID Regression Target Corruption																
η	δ_y	***CoMT***			*TI-only*		*TI+Noise*		*DUTI*		*REWLS*		*GeoMed*		*RSA*	
		Avg	Var	ϱ	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var
1.0%	10.0	**0.70**	**2.86e-5**	**0.65**	0.63	2.87e-5	0.65	2.86e-5	0.66	3.95e-5	0.64	4.26e-4	0.65	3.26e-6	0.65	2.37e-6
	20.0	**0.70**	**4.68e-6**	**0.60**	0.63	2.87e-5	0.64	4.10e-5	0.66	3.20e-5	0.64	3.03e-5	0.64	2.93e-5	0.65	4.62e-5
	30.0	**0.69**	**1.99e-5**	**0.65**	0.63	2.87e-5	0.64	2.02e-5	0.65	4.36e-6	0.62	3.40e-6	0.62	3.22e-5	0.62	3.76e-5
5.0%	10.0	**0.69**	**2.50e-5**	**0.70**	0.66	2.12e-5	0.64	2.34e-5	0.65	3.08e-5	0.64	2.96e-5	0.64	3.03e-5	0.64	3.70e-5
	20.0	**0.69**	**2.39e-5**	**0.70**	0.66	2.12e-5	0.63	2.07e-5	0.66	4.25e-5	0.65	4.77e-4	0.65	3.83e-5	0.65	4.87e-5
	30.0	**0.69**	**2.28e-5**	**0.70**	0.66	2.12e-5	0.63	2.46e-5	0.65	3.69e-6	0.63	3.64e-5	0.63	2.57e-4	0.63	3.54e-4

Table 3: Comparison of R-squared scores on *ABALONE* (**R-squared score with clean training data:**0.42)

Non-IID Regression Feature Corruption																
η	δ_x	***CoMT***			*TI-only*		*TI+Noise*		*DUTI*		*REWLS*		*GeoMed*		*RSA*	
		Avg	Var	ϱ	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var
1.0%	0.50	**0.39**	**2.05e-5**	**0.75**	0.09	3.34e-5	0.27	4.15e-5	0.29	6.51e-5	0.27	6.03e-5	0.27	5.20e-5	0.27	4.15e-5
	1.50	**0.39**	**3.64e-5**	**0.70**	0.09	3.34e-5	0.14	5.73e-5	0.15	5.20e-5	0.13	5.53e-5	0.13	4.48e-5	0.13	4.27e-5
	4.50	**0.40**	**3.54e-5**	**0.70**	0.09	3.34e-5	0.11	2.89e-5	0.12	2.53e-5	0.11	3.57e-6	0.12	5.58e-5	0.12	5.51e-5
5.0%	0.50	**0.41**	**2.73e-5**	**0.70**	0.32	3.28e-5	0.34	4.56e-5	0.36	3.23e-5	0.34	2.98e-5	0.34	4.31e-5	0.34	2.86e-5
	1.50	**0.39**	**2.27e-5**	**0.70**	0.32	3.28e-5	0.26	4.34e-5	0.29	4.56e-5	0.29	3.54e-4	0.29	2.20e-4	0.29	3.38e-5
	4.50	**0.36**	**2.28e-5**	**0.75**	0.32	3.28e-5	0.01	4.29e-5	0.10	3.83e-5	0.08	4.51e-6	0.08	3.11e-5	0.08	3.66e-5
Non-IID Regression Target Corruption																
η	δ_y	***CoMT***			*TI-only*		*TI+Noise*		*DUTI*		*REWLS*		*GeoMed*		*RSA*	
		Avg	Var	ϱ	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var
1.0%	1.50	**0.37**	**1.92e-5**	**0.75**	0.34	3.23e-5	0.31	5.00e-5	0.34	3.75e-5	0.34	4.30e-5	0.34	3.46e-5	0.34	5.46e-5
	4.50	**0.37**	**1.18e-5**	**0.75**	0.34	3.23e-5	0.31	1.25e-5	0.34	2.42e-5	0.31	2.94e-5	0.31	2.91e-5	0.31	3.72e-5
	13.50	**0.37**	**1.60e-5**	**0.75**	0.34	3.23e-5	0.31	2.90e-5	0.35	2.94e-5	0.31	3.16e-5	0.31	2.16e-5	0.31	2.32e-5
5.0%	1.50	**0.40**	**1.73e-5**	**0.70**	0.35	2.87e-5	0.33	2.42e-5	0.36	2.93e-5	0.33	3.17e-5	0.33	2.85e-5	0.33	2.54e-5
	4.50	**0.39**	**2.24e-5**	**0.70**	0.35	2.87e-5	0.33	1.68e-5	0.34	2.20e-5	0.33	3.27e-5	0.33	2.93e-5	0.31	3.07e-5
	13.50	**0.39**	**3.68e-5**	**0.70**	0.35	2.87e-5	0.32	2.67e-5	0.36	6.56e-5	0.32	6.24e-5	0.32	5.12e-5	0.32	5.54e-5

Table 4: Comparison of AUC scores on *IJCNN* (**AUC score with clean training data:**0.92)

Non-IID Classification Feature Corruption																
η	δ_x	***CoMT***			*TI-only*		*TI+Noise*		*DUTI*		*RloR*		*GeoMed*		*RSA*	
		Avg	Var	ϱ	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var
1.0%	0.50	**0.88**	**2.83e-5**	**0.65**	0.69	4.08e-5	0.71	3.12e-5	0.75	2.92e-5	0.71	4.23e-4	0.72	4.64e-6	0.72	3.94e-6
	1.50	**0.89**	**2.19e-5**	**0.70**	0.69	4.08e-5	0.62	3.06e-5	0.65	3.93e-5	0.64	4.51e-5	0.64	3.68e-5	0.64	4.27e-5
	4.50	**0.89**	**2.80e-5**	**0.70**	0.69	4.08e-5	0.64	2.71e-5	0.66	4.06e-6	0.63	4.43e-5	0.63	3.75e-5	0.63	3.16e-5
5.0%	0.50	**0.92**	**2.33e-5**	**0.70**	0.87	2.42e-5	0.73	4.91e-5	0.75	3.57e-5	0.72	4.35e-5	0.72	4.22e-5	0.72	4.28e-5
	1.50	**0.92**	**9.69e-6**	**0.70**	0.87	2.42e-5	0.58	2.18e-5	0.62	1.67e-5	0.55	1.39e-5	0.59	3.03e-5	0.59	3.46e-5
	4.50	**0.92**	**2.06e-5**	**0.75**	0.87	4.59e-5	0.54	3.65e-5	0.59	4.34e-5	0.51	4.60e-5	0.54	4.54e-5	0.54	3.69e-5
Asymmetric Label Flipping Corruption																
η	φ	***CoMT***			*TI-only*		*TI+Noise*		*DUTI*		*rLR*		*GeoMed*		*RSA*	
1.0%	0.60	**0.89**	**3.37e-5**	**0.70**	0.65	2.41e-5	0.79	2.25e-5	0.81	9.95e-5	0.73	4.30e-4	0.79	1.55e-6	0.79	1.35e-6
	0.80	**0.89**	**8.00e-6**	**0.70**	0.65	2.41e-5	0.12	4.51e-4	0.25	4.20e-5	0.11	3.37e-5	0.12	3.93e-5	0.12	3.78e-5
5.0%	0.60	**0.91**	**2.52e-5**	**0.70**	0.81	2.04e-5	0.85	3.55e-5	0.86	3.09e-5	0.83	2.82e-4	0.85	2.53e-6	0.85	3.06e-6
	0.80	**0.91**	**8.99e-6**	**0.70**	0.81	2.04e-5	0.26	3.19e-5	0.33	2.85e-5	0.24	3.69e-5	0.26	2.93e-5	0.26	2.50e-5

Table 5: Comparison of AUC scores on *SUSY* (**AUC score with clean training data:**0.79)

Non-IID Classification Feature Corruption																
η	δ_x	***CoMT***			*TI-only*		*TI+Noise*		*DUTI*		*RloR*		*GeoMed*		*RSA*	
		Avg	Var	ϱ	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var	Avg	Var
0.1%	1.50	**0.74**	**2.86e-5**	**0.75**	0.50	3.98e-5	0.68	3.21e-5	0.69	3.16e-5	0.66	3.30e-5	0.68	4.72e-5	0.68	4.99e-5
	4.50	**0.75**	**3.09e-5**	**0.70**	0.50	3.98e-5	0.56	3.48e-5	0.62	4.40e-5	0.55	3.35e-5	0.56	2.56e-6	0.56	3.35e-5
	13.50	**0.71**	**3.12e-5**	**0.60**	0.50	3.98e-5	0.51	5.04e-6	0.55	4.95e-5	0.51	3.26e-5	0.51	2.55e-5	0.51	3.26e-5
0.5%	1.50	**0.73**	**3.53e-5**	**0.60**	0.63	2.49e-5	0.65	3.21e-5	0.67	3.65e-5	0.63	4.84e-4	0.65	3.67e-5	0.65	3.58e-5
	5.50	**0.74**	**3.13e-5**	**0.70**	0.63	2.49e-5	0.64	3.60e-5	0.66	4.74e-5	0.63	6.89e-5	0.64	5.93e-5	0.64	5.88e-5
	13.50	**0.73**	**3.12e-5**	**0.70**	0.63	2.49e-5	0.50	5.05e-5	0.57	4.36e-5	0.50	4.80e-5	0.50	4.94e-05	0.50	5.26e-5
Asymmetric Label Flipping Corruption																
η	φ	***CoMT***			*TI-only*		*TI+Noise*		*DUTI*		*rLR*		*GeoMed*		*RSA*	
0.1%	0.60	**0.75**	**2.08e-5**	**0.50**	0.64	3.88e-5	0.64	4.43e-5	0.66	4.35e-5	0.63	3.43e-5	0.64	4.53e-5	0.64	4.17e-5
	0.80	**0.75**	**1.97e-5**	**0.50**	0.64	3.88e-5	0.22	3.20e-5	0.29	2.54e-4	0.22	4.20e-5	0.22	6.10e-5	0.22	4.87e-5
0.5%	0.60	**0.75**	**2.83e-5**	**0.50**	0.63	3.37e-5	0.65	3.48e-5	0.67	5.42e-5	0.62	6.52e-5	0.65	3.65e-5	0.65	3.15e-5
	0.80	**0.75**	**3.16e-5**	**0.50**	0.63	3.37e-5	0.20	3.86e-5	0.24	2.49e-5	0.20	3.30e-5	0.20	5.20e-5	0.20	5.56e-5

References

Alistarh, D.; Allen-Zhu, Z.; and Li, J. 2018. Byzantine stochastic gradient descent. In *Proceedings of NeurIPS*.

Balbach, F. J. 2008. Measuring teachability using variants of the teaching dimension. *Theoretical Computer Science* 397(1-3):94–113.

Bhahpji, A. N.; Charkraborty, S.; Mittal, P.; and Calo, S. 2018. Analyzing federated learning through an adversarial lens. In *arXiv preprint arXiv:1811.12470*.

Blanchard, P.; Mhamdi, E. M. E.; Guerraoui, R.; and Stainer, J. 2017. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Proceedings of NeurIPS*.

Bootkrajang, J., and Kaban, A. 2012. Label-noise robust logistic regression and its applications. In *ECML PKDD 2012*, 143–158.

Brückner, M., and Scheffer, T. 2011. Stackelberg games for adversarial prediction problems. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 547–555.

Chang, C.-C., and Lin, C.-J. 2011. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.* 2(3):27:1–27:27.

Chen, L.; Wang, H.; B. Charles, Z.; and Papailiopoulos, D. 2018a. Draco: Byzantine-resilient distributed training via redundant gradients. In *Proceedings of International Conference on Machine Learning*.

Chen, Y.; Singla, A.; Mac Aodha, O.; Perona, P.; and Yue, Y. 2018b. Understanding the role of adaptivity in machine teaching: The case of version space learners. *arXiv preprint arXiv:1802.05190*.

Chen, Y.; Su, L.; and Xu, J. 2017. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. In *Proceedings of ACM Conference on Measurement and Analysis of Computing Systems*, 44.

Doliwa, T.; Fan, G.; Simon, H. U.; and Zilles, S. 2014. Recursive teaching dimension, vc-dimension and sample compression. *JMLR* 15(1):3107–3131.

Fan, Y.; Tian, F.; Qin, T.; Li, X.-Y.; and Liu, T.-Y. 2018. Learning to teach. In *International Conference on Learning Representations*.

Feng, J.; Xu, H.; Mannor, S.; and Shuicheng, Y. 2014. Robust logistic regression and classification. In *Proceedings of the 27th NIPS*, 253–261.

Gervini, D., and Yohai, V. J. 2002. A class of robust and fully efficient regression. *Annual of Statistics* 30(2):583–616.

Ghosh, A.; Hong, J.; Yin, D.; and Ramchandran, K. 2019. Robust federated learning in a heterogeneous environment. *ArXiv* abs/1906.06629.

Goldman, S. A., and Kearns, M. J. 1995. On the complexity of teaching. *Journal of Computer and System Sciences* 50(1):20–31.

Haug, L.; Tschiatschek, S.; and Singla, A. 2018. Teaching inverse reinforcement learners via features and demonstrations. In *NIPS*, 8472–8481.

Jaggi, M.; Smith, V.; Takac, M.; Terhorst, J.; Krishnan, S.; Hofmann, T.; and Jordan, M. I. 2014. Communication-efficient distributed dual coordinate ascent. In *NIPS*, 3068–3076.

Karabulut, E. 2017. Distributed integer programming. *Phd Thesis, GeorgiaTech*.

Li, L.; Xu, W.; Chen, T.; Giannakis, G.; and Ling, Q. 2018. Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. In *Proceedings of AAAI*.

Liu, W.; Dai, B.; Humayun, A.; Tay, C.; Yu, C.; Smith, L. B.; Rehg, J. M.; and Song, L. 2017a. Iterative machine teaching. In *ICML*, 2149–2158.

Liu, W.; Dai, B.; Li, X.; Liu, Z.; Rehg, J. M.; and Song, L. 2017b. Towards black-box iterative machine teaching. *arXiv preprint arXiv:1710.07742*.

Liu, J.; Zhu, X.; and Ohannessian, H. 2016. The teaching dimension of linear learners. In *ICML*, 117–126.

Ma, Y.; Nowak, R.; Rigollet, P.; Zhang, X.; and Zhu, X. 2018. Teacher improves learning by selecting a training subset. In *International Conference on Artificial Intelligence and Statistics*, 1366–1375.

Rahimi, A., and Recht, B. 2007. Random features for large-scale kernel machines. In *NIPS*, 143–158.

Smith, V.; Forte, S.; Ma, C.; Takáč, M.; Jordan, M. I.; and Jaggi, M. 2018. Cocoa: A general framework for communication-efficient distributed optimization. *JMLR* 19:1–49.

Su, L., and Xu, J. 2019. Securing distributed gradient descent in high dimensional statistical learning. *Proceedings of ACM Measurement Analysis and Computer System* 3:12:1–12:41.

Xie, C.; Koyejo, O.; and Gupta, I. 2018a. Generalized byzantine-tolerant sgd. *ArXiv* abs/1802.10116.

Xie, C.; Koyejo, O.; and Gupta, I. 2018b. Phocas: dimensional byzantine-resilient stochastic gradient descent. *ArXiv* abs/1805.09682.

Yang, Q.; Liu, Y.; Chen, T.; and Tong, Y. 2019. Federated machine learning: Concept and applications. *ACM Transaction on Intelligent Systems and Technology* 10(2).

Yin, D.; Chen, Y.; Kannan, R.; and Bartlett, P. 2018. Byzantine-robust distributed learning: Towards optimal statistical rates. In *Proceedings of the 35th International Conference on Machine Learning*, 5650–5659.

Zhang, X.; Zhu, X.; and Wright, S. 2018. Training set debugging using trusted items. In *AAAI*.

Zhu, J. 2013. Machine teaching for bayesian learners in the exponential family. In *NIPS*, 1905–1913.

Zilles, S.; Lange, S.; Holte, R.; and Zinkevich, M. 2011. Models of cooperative teaching and learning. *JMLR* 12(Feb):349–384.