CS229 Project Report: An Efficient Algorithm for Robust Collaborative Learning

Mingda Qiao
mqiao@stanford.edu

1 Introduction

The *collaborative learning* framework proposed by [1] models the situation where one tries to learn the ground truth from a large and diverse crowd. For instance, suppose we are training a spam filter for millions of email users. While the users share the same target function (i.e., whether an email is a spam), their data distributions are heterogeneous—they receive emails from different addresses, written in different languages and styles. The amount of data provided by each individual user is far from sufficient for training an accurate model, yet naïvely combining the datasets could also be problematic—the learned model is only guaranteed to work well on average, yet it can be unacceptably inaccurate for some users.

What complicates the situation is the possible existence of outliers and adversaries. As one employs the learning system for a large number of users, it is conceivable that a small fraction of the users might not provide the system with accurate training examples, and might even sabotage the learner maliciously. The *robust collaborative learning* model proposed by [2] takes into account the existence of adversarial users, and studies whether efficient collaboration is still possible in this setting.

It is shown in [2] that there is an algorithm that achieves an $O(m+\log n)$ *overhead* when there are n users in total and at most m of them are adversarial. Here the overhead, roughly speaking, is the multiplicative increase in the number of required samples compared to the single user case. However, the proposed algorithm involves an exhaustive search over subsets of the users, which renders the method computationally inefficient, and thus impractical for any interesting settings.

The robust collaborative learning setting is also closely related to the literature on multi-task learning and robust statistics; we refer the reader to the references in [1, 2] due to space limits.

2 Theoretical Results

2.1 Model

We start by summarizing the *robust collaborative model* proposed by [2]. The model considers a binary classification task over space $\mathcal{X}$ with hypothesis class $\mathcal{F}$, i.e., $\mathcal{F}$ is a family of functions mapping $\mathcal{X}$ to $\{0,1\}$. For simplicity, we focus on the *realizable* case, where there is a ground truth function $f^* \in \mathcal{F}$. There are n users, each associated with a data distribution $\mathcal{D}_i$ over $\mathcal{X}$. The learning algorithm interacts with the users by (possibly adaptively) collecting labeled data from them. Upon each request, a *truthful user* draws a data point x from its distribution $\mathcal{D}_i$, and returns the labeled pair $(x, f^*(x))$. On the other hand, the output of an *adversarial user* can be an arbitrary pair from $\mathcal{X} \times \{0,1\}$.

The output of the learning algorithm is a sequence of n functions $f_1, f_2, \ldots, f_n$. The function sequence is considered as ϵ-correct if its accuracy is at least $1-\epsilon$ for each truthful user's data distribution, i.e., for each truthful user i, it holds that

$$\Pr_{x\sim\mathcal{D}_i}\left[f_i(x) = f^*(x)\right] \ge 1-\epsilon.$$

2.2 An efficient algorithm

Our newly proposed meta-algorithm is shown in the following figure. The $\tilde{\Theta}$ notation hides a universal constant factor and also factors that are logarithmic in $1/\epsilon$.

Algorithm 1: Robust collaborative learning from user sampling.

```
Input: Hypothesis class $\mathcal{F}$ with VC-dimension $d$. Parameters $n$, $m$, $\epsilon$.
Output: Functions $f_1, f_2, \ldots, f_n$.
1  $G \leftarrow [n]$;
2  while $m/|G| < 1/10$ do
3      repeat
4          $G' \leftarrow$ random subset of $G$ with size $\frac{|G|}{2m}$;
5          Collect $\tilde{\Theta}(d/\epsilon)$ samples from the mixture $\frac{1}{|G'|}\sum_{i \in G'} \mathcal{D}_i$ to obtain dataset $S$;
6      until there exists $\hat{f} \in \mathcal{F}$ consistent with $S$;
7      for each user $i \in G'$ do
8          Collect $\tilde{\Theta}(1/\epsilon)$ samples from user $i$;
9          if $\hat{f}$ achieves $> 1 - \epsilon/2$ accuracy on the collected data then
10             $f_i \leftarrow \hat{f}$;
11             $G \leftarrow G \setminus \{i\}$;
12 for each remaining user $i \in G$ do
13     Collect $\tilde{\Theta}(d/\epsilon)$ samples from user $i$;
14     Find $\hat{f} \in \mathcal{F}$ that is consistent with the collected data;
15     $f_i \leftarrow \hat{f}$;
```

In words, the algorithm maintains a set G of "active users" and proceeds as follows:

1. As long as the fraction of adversarial users (among active users) is below a constant (1/10), sample a $\Theta(1/m)$ fraction of the active users, and draw $\tilde{\Theta}(d/\epsilon)$ training examples from their mixture distribution. Find a function $\hat{f}$ that is consistent with the data. If no such function exists, repeat this step.

2. Then, as a validation step, sample $\tilde{\Theta}(1/\epsilon)$ data points from each of the sampled users to verify that $\hat{f}$ has accuracy $1 - \epsilon$ on them. For each user that passes the above verification, select $\hat{f}$ as the answer for that user, and mark the user as inactive.

3. Repeat Steps 1 and 2 until the fraction of adversarial users is above a constant. This means that at most $O(m)$ users are left. In this case, simply learn an ϵ-accurate function for each of the users separately, using $\tilde{\Theta}(d/\epsilon)$ samples per user.

Sketch of the analysis. We first show that the first step (the repeat-loop) will not be repeated too many times to make progress. Indeed, one can show that with probability $\Omega(1)$, all the $|G|/(2m)$ sampled users in G' are truthful. Then, the ground truth f^* would be consistent with the data points. Thus, the repeat-loop will terminate after $O(1)$ repetitions in expectation.

Then, we argue that the outer while-loop will be repeated at most $O(m \log n)$ times. After the first step, since $\hat{f}$ is consistent with the $\tilde{\Theta}(d/\epsilon)$ labeled data points, by the VC theory, it achieves an $O(\epsilon)$ error on the *mixture distribution* defined by the users. By an averaging argument, it would be $O(\epsilon)$-accurate for a constant fraction of the $|G|/(2m)$ active users. Thus, we can satisfy a $\Theta(1/m)$ fraction of the active users. Since $\left(1 - \frac{1}{m}\right)^{m \ln n} \le \exp\left(-\frac{1}{m} \cdot m \ln n\right) = \frac{1}{n}$, the while-loop will eventually terminate in $O(m \log n)$ rounds.

Putting everything together, each of the $O(m \log n)$ while-loop iterations takes $\tilde{\Theta}(d/\epsilon)$ samples, and each of the $O(m)$ remaining users in G takes $\tilde{\Theta}(d/\epsilon)$ samples in the end. Thus, the sample complexity of

Algorithm 1 is $\tilde{O}(\frac{md\log n}{\epsilon})$. Recall from the VC theory that $\tilde{\Theta}(d/\epsilon)$ samples are sufficient and necessary for the single-user case. Thus, Algorithm 1 achieves an overhead of $O(m\log n)$, matching the $\Theta(m+\log n)$ bound in [2] up to a logarithmic factor.

3 Empirical Results

We evaluate the empirical performance of the proposed Algorithm 1 and comparing it with those of the naïve baselines.

3.1 Setup

Classification tasks. We will focus on the following two simple classification tasks: linear classification over $\mathbb{R}^d$ and parity learning over $\{0,1\}^d$. For each linear regression (resp. parity learning) instance, a ground truth vector $\theta^* \in \mathbb{R}^d$ (resp. $\theta^* \in \{0,1\}^d$) will be sampled randomly at the beginning.

Data distribution. Each user will be assigned a data distribution, which is supported on a subspace of $\mathbb{R}^d$ (resp. $\{0,1\}^d$) of dimension $d_0 < d$. This models the aspect of the motivating example that each data source might only provides information of the ground truth "in certain directions". In particular, we consider the following two model for the dimension of subspaces:

1. "Random": Each user samples d_0 from $[d]$ uniformly and independently at random.
2. "Pivot": One of the truthful users ("pivot user") chooses $d_0 = d$ while all the other users have $d_0 = 1$. This models the case where only one of the n data sources are informative.

Adversarial model. m adversarial users will be sampled uniformly from the n users. While a truthful user provides correctly labeled samples, an adversarial user would sample the instance x from his data distribution, yet provide the learner with the opposite label.

Parameter setting. We run the experiments on a synthetic setting defined as above with $n = 200$ users and m adversaries. The dimension of the instance space is set to $d = 500$.

3.2 Methods and Evaluation

We will evaluate the following approaches to robust collaborative learning:

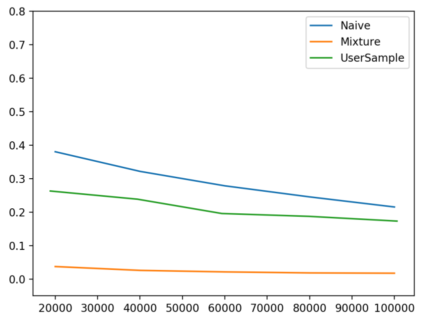
1. Naïve: Learn an ϵ-accurate function for each of the n users separately. Note that this approach has a trivial $\Theta(n)$ overhead.
2. Mixture: Directly learn an ϵ-accurate classifier from the mixture distribution of all data sources, and apply the learned model to all users. Note that even when no adversaries are present, this approach only guarantees that the learned classifier is ϵ-accurate *on the mixture distribution* instead of the individual data distributions. As a result, the outcome could have a high error on a small fraction of the users, which is not acceptable.
3. UserSample: The proposed algorithm shown in Algorithm 1.

We also note that the algorithm proposed in [2] is not computationally feasible in the setting, as it involves a search over nearly $2^n = 2^{200}$ subsets of users, which is far from being practical.

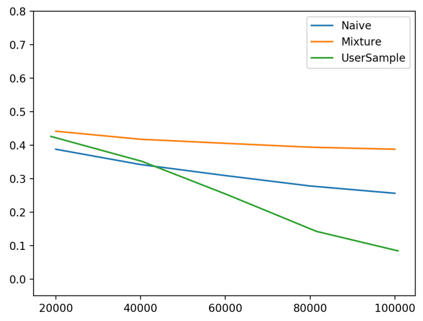
The above methods will be evaluated in two criteria: the number of samples and the worst-case accuracy, i.e., the highest test error among all the truthful users. Specifically, since Algorithm 1 is among those "theorists' algorithms" and are mostly of theoretical value, some of the constant factors are either not optimized or unspecified. To make meaningful comparisons possible, we will tune the parameters in Algorithm 1 as well as the naïve baselines, and plot the tradeoff curve between the number of samples and the resulting worst-case error.

3.3 Results

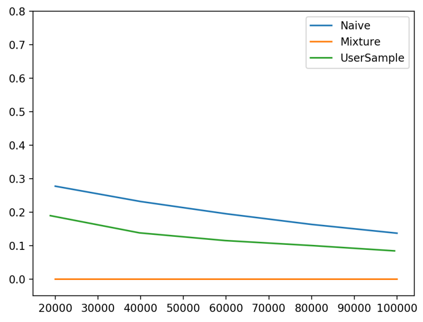
Our results on synthetic data are shown in Figure 1. The proposed UserSample algorithm is more sample-efficient than the two baselines when the dimensions of the subspaces are chosen as the Pivot setting. When the dimensionality of the subspaces are chosen more uniformly (in the Random setting), it turns out that the Mixture baseline is more efficient than UserSample.



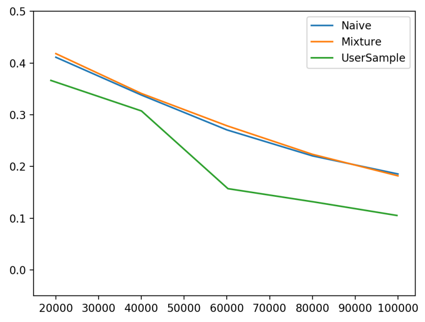
(a) linear function, random subspaces



(b) linear function, pivot subspaces



(c) parity function, random subspaces



(d) parity function, pivot subspaces

Figure 1: The X-axis denotes the average number of samples requested by the algorithm. The Y-axis denotes the average worst-case error among all truthful users. Both quantities are averaged over 10 trials for the same parameter setting.

4 Future Work

This project leaves a few interesting open questions from on theoretical and empirical sides, which I will briefly outline below.

Theoretical questions. The obvious open problem is to resolve the gap between the current $O(m \log n)$ overhead and the optimal $\Theta(m + \log n)$ bound. One naïve approach to improving the UserSample is to sample a larger group of active users so that each iteration makes more progress. It turns out that this would significantly decrease the success probability of each iteration and end up with spending much more samples.

Apart from the question above, another direction would be designing specialized algorithms for specific hypothesis classes, e.g., the class of linear classifiers. While it seems hard to match the optimal overhead efficiently in the general binary classification setting, it might worth working with specific, structured learning tasks including linear classification and parity learning.

Empirical questions. While the proposed algorithm is sample-efficient from a theoretical perspective, its empirical performance is not significantly better (if not worse) than the naïve baselines, especially when the user distributions are generated randomly. The current work leaves the open question whether there is a more efficient implementation of the theoretically-principled algorithm, as well as whether the algorithm could be adapted so that its performance on nicely-structured instances become comparable to that of the simple baselines.

5 Contribution and Project Code

This is a single-author project and all the aforementioned results are my own work. The code for the experiments can be found at:

`https://drive.google.com/file/d/1OZ5XnXjXt0ej_IexG4UT_l2VtYxZs72E`

References

[1] Avrim Blum, Nika Haghtalab, Ariel D. Procaccia, and Mingda Qiao. Collaborative PAC learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[2] Mingda Qiao. Do outliers ruin collaboration? In *International Conference on Machine Learning (ICML)*, pages 4180–4187, 2018.