Learning from Data with Heterogeneous Noise using SGD

Shuang Song
University of California, San Diego

Kamalika Chaudhuri
University of California, San Diego

Anand D. Sarwate
Rutgers University

Abstract

We consider learning from data of variable quality that may be obtained from different heterogeneous sources. Addressing learning from heterogeneous data in its full generality is a challenging problem. In this paper, we adopt instead a model in which data is observed through heterogeneous noise, where the noise level reflects the quality of the data source. We study how to use stochastic gradient algorithms to learn in this model. Our study is motivated by two concrete examples where this problem arises naturally: learning with local differential privacy based on data from multiple sources with different privacy requirements, and learning from data with labels of variable quality.

The main contribution of this paper is to identify how heterogeneous noise impacts performance. We show that given two datasets with heterogeneous noise, the order in which to use them in standard SGD depends on the learning rate. We propose a method for changing the learning rate as a function of the heterogeneity, and prove new regret bounds for our method in two cases of interest. Finally, we evaluate the performance of our algorithm on real data.

1 INTRODUCTION

Modern large-scale machine learning systems often integrate data from several different sources. In many cases, these sources provide data of a similar type (i.e. with the same features) but collected under different circumstances. For example, patient records from different studies of a particular drug may be combined to perform a more comprehensive analysis, or a collection of images with annotations from experts as well as non-experts may be combined to learn a predictor. In particular, data from different sources may be of varying *quality*. In this paper we adopt a model in which data is observed through heterogeneous noise, where the noise level reflects the quality of the data source. We study how to use stochastic gradient algorithms to learn from data of heterogeneous quality.

In full generality, learning from heterogeneous data is essentially the problem of domain adaptation – a challenge for which good and complete solutions are difficult to obtain. Instead, we focus on the special case of heterogeneous noise and show how to use information about the data quality to improve the performance of learning algorithms which ignore this information.

Two concrete instances of this problem motivate our study: locally differentially private learning from multiple sites, and classification with random label noise. Differential privacy (Dwork et al., 2006b,a) is a privacy model that has received significant attention in machine-learning and data-mining applications. A variant of differential privacy is *local privacy* – the learner can only access the data via noisy estimates, where the noise guarantees privacy (Duchi et al., 2012, 2013). In many applications, we are required to learn from sensitive data collected from individuals with heterogeneous privacy preferences, or from multiple sites with different privacy requirements; this results in the heterogeneity of noise added to ensure privacy. Under random classification noise (RCN) (Kearns, 1998), labels are randomly flipped before being presented to the algorithm. The heterogeneity in the noise addition comes from combining labels of variable quality – such as labels assigned by domain experts with those assigned by a crowd.

To our knowledge, Crammer et al. (2006) were the first to provide a theoretical study of how to learn classifiers from data of variable quality. In their formulation, like ours, data is observed through heterogeneous noise. Given data with known noise levels, their study focuses on finding an optimal ordering of the data and a stopping rule without any constraint on the compu-

Appearing in Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS) 2015, San Diego, CA, USA. JMLR: W&CP volume 38. Copyright 2015 by the authors.

tational complexity. We instead shift our attention to studying *computationally efficient strategies* for learning classifiers from data of variable quality.

We propose a model for variable data quality which is natural in the context of large-scale learning using stochastic gradient descent (SGD) and its variants (Bottou, 2010; Bekkerman et al., 2011). We assume that the training data are accessed through an oracle which provides an unbiased but noisy estimate of the gradient of the objective. The noise comes from two sources: the random sampling of a data point, and additional noise due to the data quality. Our two motivating applications – learning with local differential privacy and learning from data of variable quality – can both be modeled as solving a regularized convex optimization problem using SGD. Learning from data with heterogeneous noise in this framework thus reduces to running SGD with noisy gradient estimates, where the magnitude of the added noise varies across iterations.

Main results. In this paper we study noisy stochastic gradient methods when learning from multiple data sets with different noise levels. For simplicity we consider the case where there are two data sets, which we call Clean and Noisy. We process these data sets sequentially using SGD with learning rate $\mathcal{O}(1/t)$. In a future full version of this work we also analyze averaged gradient descent (AGD) with learning rate $\mathcal{O}(1/\sqrt{t})$. We address some basic questions in this setup:

In what order should we process the data? Suppose we use standard SGD on the union of Clean and Noisy. We show theoretically and empirically that the order in which we should process the datasets to get good performance depends on the learning rate of the algorithm: in some cases we should use the order (Clean, Noisy) and in others (Noisy, Clean).

Can we use knowledge of the noise rates? We show that using separate learning rates that depend on the noise levels for the clean and noisy datasets improves the performance of SGD. We provide a heuristic for choosing these rates by optimizing an upper bound on the error for SGD that depends on the ratio of the noise levels. We analytically quantify the performance of our algorithm in two regimes of interest. For moderate noise levels, we demonstrate empirically that our algorithm outperforms using a single learning rate and using clean data only.

Does using noisy data always help? The work of Crammer et al. (2006) suggests that if the noise level of noisy data is above some threshold, then noisy data will not help. Moreover, when the noise levels are very high, our heuristic does not always empirically outperform simply using the clean data. On the other hand, our theoretical results suggest that changing the learning rate can make noisy data useful. How do we resolve this apparent contradiction?

We perform an empirical study to address this question. Our experiments demonstrate that very often, there exists a learning rate at which noisy data helps; however, because the actual noise level may be far from the upper bound used in our algorithm, our optimization may not choose the best learning rate for every data set. We demonstrate that by adjusting the learning rate we can still take advantage of noisy data.

For simplicity we, like previous work Crammer et al. (2006), assume that the algorithms know the noise levels exactly. However, our algorithms can still be applied in the presence of approximate knowledge of the noise levels, and our result on the optimal data order only needs to know which dataset has more noise.

Related Work. There has been significant work on the convergence of SGD assuming analytic properties of the objective function, such as strong convexity and smoothness. When the objective function is λ-strongly convex, the learning rate used for SGD is $\mathcal{O}(1/\lambda t)$ (Nemirovsky and Yudin, 1983; Agarwal et al., 2009; Rakhlin et al., 2012; Bach and Moulines, 2011), which leads to a regret of $\mathcal{O}(1/\lambda^2 t)$ for smooth objectives. For non-smooth objectives, SGD with learning rate $\mathcal{O}(1/\lambda t)$ followed by some form of averaging of the iterates achieves $\mathcal{O}(1/\lambda t)$ (Nesterov and Vial, 2008; Nemirovski et al., 2009; Shalev-Shwartz et al., 2009; Xiao, 2010; Duchi and Singer, 2009).

There is also a body of literature on differentially private classification by regularized convex optimization in the batch (Chaudhuri et al., 2011; Rubinstein et al., 2013; Kifer et al., 2012) as well as the online (Jain et al., 2012) setting. In this paper, we consider classification with *local differential privacy* (Wasserman and Zhou, 2010; Duchi et al., 2012), a stronger form of privacy than ordinary differential privacy. Duchi et al. (2012) propose learning a classifier with local differential privacy using SGD, and Song et al. (2013) show empirically that using mini-batches significantly improves the performance of differentially private SGD. Recent work by Bassily et al. (2014) provides an improved privacy analysis for non-local privacy. Our work is an extension of these papers to heterogeneous privacy requirements.

Crammer et al. (2006) study classification when the labels in each data set are corrupted by RCN of different rates. Assuming the classifier minimizing the empirical 0/1 classification error can always be found, they propose a general theoretical procedure that processes the datasets in increasing order of noise, and determines when to stop using more data. In contrast, our noise model is more general and we provide a polynomial time algorithm for learning. Our results imply that in some cases the algorithm should process the noisy data first, and finally, our algorithm uses all the data.

2 THE MODEL

We consider linear classification in the presence of noise. We are given T labelled examples $(x_1, y_1), \ldots, (x_T, y_T)$, where $x_i \in \mathbb{R}^d$, and $y_i \in \{-1, 1\}$ and our goal is to find a hyperplane w that largely separates the examples labeled 1 from those labeled -1. A standard solution is via the following regularized convex optimization problem:

$$w^* = \operatorname*{\mathbf{argmin}}_{w \in \mathcal{W}} f(w) := \frac{\lambda}{2}\|w\|^2 + \frac{1}{T}\sum_{i=1}^{T} \ell(w, x_i, y_i). \tag{1}$$

Here ℓ is a convex loss function, and $\frac{\lambda}{2}\|w\|^2$ is a regularization term. Popular choices for ℓ include the logistic loss $\ell(w, x, y) = \log(1 + e^{-yw^\top x})$ and the hinge loss $\ell(w, x, y) = \max(0, 1 - yw^\top x)$.

Stochastic Gradient Descent (SGD) is a popular approach to solving (1): starting with an initial w_1, at step t, SGD updates w_{t+1} using the point (x_t, y_t) as follows:

$$w_{t+1} = \Pi_{\mathcal{W}} \left(w_t - \eta_t(\lambda w_t + \nabla \ell(w_t, x_t, y_t))\right). \tag{2}$$

Here Π is a projection operator onto the convex feasible set $\mathcal{W}$, typically set to $\{w : \|w\|_2 \leq 1/\lambda\}$ and η_t is a learning rate (or step size) which specifies how fast w_t changes. A common choice for the learning rate for the case when $\lambda > 0$ is c/t, where $c = \Theta(1/\lambda)$.

2.1 The Heterogeneous Noise Model

We propose an abstract model for heterogeneous noise that can be specialized to two important scenarios: differentially private learning, and random classification noise. By heterogeneous noise we mean that the distribution of the noise can depend on the data points themselves. More formally, we assume that the learning algorithm may only access the labeled data through an oracle $\mathcal{G}$ which, given a $w \in \mathbb{R}^d$, draws a fresh independent sample (x, y) from the underlying data distribution, and returns an unbiased noisy gradient of the objective function $\nabla f(w)$, based on the example (x, y):

$$\mathbb{E}\left[\mathcal{G}(w)\right] = \lambda w + \nabla \ell(w, x, y),\ \mathbb{E}\left[\|\mathcal{G}(w)\|^2\right] \leq \Gamma^2. \tag{3}$$

The precise manner in which $\mathcal{G}(w)$ is generated depends on the application. Define the *noise level* for the oracle $\mathcal{G}$ as the constant Γ in (3); larger Γ means more noisy data. Finally, to model finite training datasets, we assume that an oracle $\mathcal{G}$ may be called only a limited number of times.

Observe that in this noise model, we can easily use the noisy gradient returned by $\mathcal{G}$ to perform SGD. The update rule becomes:

$$w_{t+1} = \Pi_{\mathcal{W}} \left(w_t - \eta_t \mathcal{G}(w_t)\right). \tag{4}$$

The SGD estimate is w_{t+1}.

In practice, we can implement an oracle such as $\mathcal{G}$ based on a finite labelled training set D as follows. We apply a random permutation on the samples in D, and at each invocation, compute a noisy gradient based on the next sample in the permutation. The number of calls to the oracle is limited to $|D|$. If the samples in D are drawn iid from the underlying data distribution, and if any extraneous noise added to the gradient at each iteration is unbiased and drawn independently, then this process will implement the oracle correctly.

To model heterogeneous noise, we assume that we have access to two oracles $\mathcal{G}_1$ and $\mathcal{G}_2$ implemented based on datasets D_1 and D_2, which can be called at most $|D_1|$ and $|D_2|$ times respectively. For $j = 1, 2$, the noise level of oracle $\mathcal{G}_j$ is Γ_j, and the values of Γ_1 and Γ_2 are known to the algorithm. In some practical situations, Γ_1 and Γ_2 will not be known exactly; however, our algorithm in Section 4 also applies when approximate noise levels are known, and our algorithm in Section 3 applies even when only the relative noise levels are known.

2.1.1 Local Differential Privacy

Local differential privacy (Wasserman and Zhou, 2010; Duchi et al., 2012; Kasiviswanathan et al., 2008) is a strong notion of privacy motivated by differential privacy (Dwork et al., 2006b). An untrusted algorithm is allowed to access a perturbed version of a sensitive dataset through a sanitization interface, and must use this perturbed data to perform some estimation. The amount of perturbation is controlled by a parameter ϵ, which measures the privacy risk.

Definition 1 (Local Differential Privacy). *Let $D = (X_1, \ldots, X_n)$ be a sensitive dataset where each $X_i \in \mathcal{D}$ corresponds to data about individual i. A randomized sanitization mechanism M which outputs a disguised version $(U_1, \ldots, U_n)$ of D is said to provide ϵ-local differential privacy to individual i, if for all $x, x' \in D$ and for all $S \subseteq \mathcal{S}$,*

$$\Pr(U_i \in S | X_i = x) \leq e^{\epsilon} \Pr(U_i \in S | X_i = x'). \tag{5}$$

Here the probability is taken over the randomization in the sanitization mechanism, and ϵ is a parameter that measures privacy risk where smaller ϵ means less privacy risk.

Consider learning a linear classifier from a sensitive labelled dataset while ensuring local privacy of the participants. This problem can be expressed in our noise model by setting the sanitization mechanism as the oracle. Given a privacy risk ϵ, for $w \in \mathbb{R}^d$, the oracle $\mathcal{G}^{\text{DP}}$ draws a random labelled sample (x, y) from the underlying data distribution, and returns the noisy gradient of the objective function at w computed based on (x, y) as

$$\mathcal{G}^{\text{DP}}(w) = \lambda w + \nabla \ell(w, x, y) + Z, \qquad (6)$$

where Z is independent random noise drawn from the density: $\rho(z) \propto e^{-(\epsilon/2)\|z\|}$.

Duchi et al. (2012) showed that this mechanism provides ϵ-local privacy assuming analytic conditions on the loss function, bounded data, and that the oracle generates a fresh random sample at each invocation. The following result shows how to set the parameters to fit in our heterogeneous noise model. The proof is provided in the supplement.

Theorem 1. *If $\|\nabla \ell(w, x, y)\| \leq 1$ for all w and (x, y), then $\mathcal{G}^{DP}(w)$ is ϵ-local differentially private. Moreover, for any w such that $\|w\| \leq \frac{1}{\lambda}$, $\mathbb{E}[\mathcal{G}^{DP}(w)] = \lambda w + \nabla \mathbb{E}_{(x,y)}[\ell(w, x, y)]$, and*

$$\mathbb{E}[\|\mathcal{G}^{DP}(w)\|^2] \leq 4 + \frac{4(d^2 + d)}{\epsilon^2}.$$

In practice, we may wish to learn classifiers from multiple sensitive datasets with different privacy parameters. For example, suppose we wish to learn a classifier from sensitive patient records in two different hospitals holding data sets D_1 and D_2, respectively. The hospitals have different privacy policies, and thus different privacy parameters ϵ_1 and ϵ_2. This corresponds to a heterogeneous noise model in which we have two sanitizing oracles – $\mathcal{G}_1^{\text{DP}}$ and $\mathcal{G}_2^{\text{DP}}$. For $j = 1, 2$, $\mathcal{G}_j^{\text{DP}}$ implements a differentially private oracle with privacy parameter ϵ_j based on dataset D_j and may be called at most $|D_j|$ times.

2.1.2 Random Classification Noise

In the random classification noise model of Kearns (1998), the learning algorithm is presented with labelled examples $(x_1, \tilde{y}_1), \ldots, (x_T, \tilde{y}_T)$, where each $\tilde{y}_i \in \{-1, 1\}$ has been obtained by independently flipping the *true label* y_i with some probability σ. Natarajan et al. (2013) showed that solving

$$\underset{w}{\textbf{argmin}} \frac{\lambda}{2}\|w\|^2 + \frac{1}{T}\sum_{i=1}^{T} \tilde{\ell}(w, x_i, \tilde{y}_i, \sigma) \qquad (7)$$

yields a linear classifier from data with random classification noise, where $\tilde{\ell}$ is a surrogate loss function corresponding to a convex loss ℓ:

$$\tilde{\ell}(w, x, y, \sigma) = \frac{(1-\sigma)\ell(w, x, y) - \sigma \ell(w, x, -y)}{1 - 2\sigma},$$

and σ is the probability that each label is flipped. This problem can be expressed in our noise model using an oracle $\mathcal{G}^{\text{RCN}}$ which on input w draws a fresh labelled example $(x, \tilde{y})$ and returns

$$\mathcal{G}^{\text{RCN}}(w) = \lambda w + \nabla \tilde{\ell}(w, x, \tilde{y}, \sigma).$$

The SGD updates in (4) with respect to $\mathcal{G}^{\text{RCN}}$ minimize (7). If $\|x\| \leq 1$ and $\|\nabla \ell(w, x, y)\| \leq 1$, we have $\mathbb{E}[\mathcal{G}_{\text{RCN}}(w)] = \lambda w + \nabla \ell(w, x, y)$ and $\mathbb{E}\left[\|\mathcal{G}^{\text{RCN}}(w)\|_2^2\right] \leq 3 + 1/(1-2\sigma)^2$, under the random classification noise assumption, so the oracle $\mathcal{G}^{\text{RCN}}$ satisfies the conditions in (3) with $\Gamma^2 = 3 + 1/(1-2\sigma)^2$.

In practice, we may wish to learn classifiers from multiple datasets with different amounts of classification noise (Crammer et al., 2006); for example, we may have a small dataset D_1 labeled by domain experts, and a larger noisier dataset D_2, labeled via crowdsourcing, with flip probabilities σ_1 and σ_2. We model this scenario using two oracles – $\mathcal{G}_1^{\text{RCN}}$ and $\mathcal{G}_2^{\text{RCN}}$. For $j = 1, 2$, oracle $\mathcal{G}_j^{\text{RCN}}$ is implemented based on D_j and flip probability σ_j, and may be called at most $|D_j|$ times.

3 DATA ORDER DEPENDS ON LEARNING RATE

Suppose we have two oracles $\mathcal{G}_{\mathsf{C}}$ (for "clean") and $\mathcal{G}_{\mathsf{N}}$ (for "noisy") implemented based on datasets $D_{\mathsf{C}}, D_{\mathsf{N}}$ with noise levels $\Gamma_{\mathsf{C}}, \Gamma_{\mathsf{N}}$ (where $\Gamma_{\mathsf{C}} < \Gamma_{\mathsf{N}}$) respectively. In which order should we query the oracle when using SGD? Perhaps surprisingly, it turns out that the answer depends on the learning rate. Below, we show a specific example of a convex optimization problem such that with $\eta_t = c/t$, the optimal ordering is to use $\mathcal{G}_{\mathsf{C}}$ first when $c \in (0, 1/\lambda)$, and the optimal ordering is to use $\mathcal{G}_{\mathsf{N}}$ first when $c > 1/\lambda$.

Let $|D_{\mathsf{C}}| + |D_{\mathsf{N}}| = T$ and consider the convex optimization problem:

$$\min_{w \in \mathcal{W}} \frac{\lambda}{2}\|w\|^2 - \frac{1}{T}\sum_{i=1}^{T} y_i w^\top x_i, \qquad (8)$$

where the points $\{(x_i, y_i)\}$ are drawn from the underlying distribution by $\mathcal{G}_{\mathsf{C}}$ or $\mathcal{G}_{\mathsf{N}}$. Suppose $\mathcal{G}(w) = \lambda w - yx + Z$ where Z is an independent noise vector such that $\mathbb{E}[Z] = 0$, $\mathbb{E}[\|Z\|^2] = V_{\mathsf{C}}^2$ if $\mathcal{G}$ is $\mathcal{G}_{\mathsf{C}}$, and $\mathbb{E}[\|Z\|^2] = V_{\mathsf{N}}^2$ if $\mathcal{G}$ is $\mathcal{G}_{\mathsf{N}}$ with $V_{\mathsf{N}}^2 \geq V_{\mathsf{C}}^2$.

For our example, we consider the following three variants of SGD: CF and NF for "clean first" and "noisy first" and AO for an "arbitrary ordering":

1. CF: For $t \leq |D_{\mathsf{C}}|$, query $\mathcal{G}_{\mathsf{C}}$ in the SGD update (4). For $t > |D_{\mathsf{C}}|$, query $\mathcal{G}_{\mathsf{N}}$.
2. NF: For $t \leq |D_{\mathsf{N}}|$, query $\mathcal{G}_{\mathsf{N}}$ in the SGD update (4). For $t > |D_{\mathsf{N}}|$, query $\mathcal{G}_{\mathsf{C}}$.
3. AO: Let S be an arbitrary sequence of length T consisting of $|D_{\mathsf{C}}|$ C's and $|D_{\mathsf{N}}|$ N's. In the SGD update (4) in round t, if the t-th element S_t of S is C, then query $\mathcal{G}_{\mathsf{C}}$; else, query $\mathcal{G}_{\mathsf{N}}$.

In order to isolate the effect of the noise, we consider two additional oracles $\mathcal{G}'_{\mathsf{C}}$ and $\mathcal{G}'_{\mathsf{N}}$; the oracle $\mathcal{G}'_{\mathsf{C}}$ (resp. $\mathcal{G}'_{\mathsf{N}}$) is implemented based on the dataset D_{C} (resp. D_{N}), and iterates over D_{C} (resp. D_{N}) in exactly the same order as $\mathcal{G}_{\mathsf{C}}$ (resp. $\mathcal{G}_{\mathsf{N}}$); the only difference is that for $\mathcal{G}'_{\mathsf{C}}$ (resp. $\mathcal{G}'_{\mathsf{N}}$), no extra noise is added to the gradient (that is, $Z = 0$). The main result of this section is stated in Theorem 2.

Theorem 2. *Let $\{w_t^{\mathsf{CF}}\}$, $\{w_t^{\mathsf{NF}}\}$ and $\{w_t^{\mathsf{AO}}\}$ be the sequences of updates obtained by running SGD for objective function* (8) *under CF, NF and AO respectively, and let $\{v_t^{\mathsf{CF}}\}$, $\{v_t^{\mathsf{NF}}\}$ and $\{v_t^{\mathsf{AO}}\}$ be the sequences of updates under CF, NF and AO with calls to $\mathcal{G}_{\mathsf{C}}$ and $\mathcal{G}_{\mathsf{N}}$ replaced by calls to $\mathcal{G}'_{\mathsf{C}}$ and $\mathcal{G}'_{\mathsf{N}}$. Let $T = |D_{\mathsf{C}}| + |D_{\mathsf{N}}|$.*

1. *If the learning rate $\eta_t = c/t$ where $c \in (0, \frac{1}{\lambda})$, then* $\mathbb{E}\left[\|v_{T+1}^{\mathsf{CF}} - w_{T+1}^{\mathsf{CF}}\|^2\right] \leq \mathbb{E}\left[\|v_{T+1}^{\mathsf{AO}} - w_{T+1}^{\mathsf{AO}}\|^2\right]$.
2. *If the learning rate $\eta_t = c/t$ where $c > 1/\lambda$, then* $\mathbb{E}\left[\|v_{T+1}^{\mathsf{NF}} - w_{T+1}^{\mathsf{NF}}\|^2\right] \leq \mathbb{E}\left[\|v_{T+1}^{\mathsf{AO}} - w_{T+1}^{\mathsf{AO}}\|^2\right]$.

This means arbitrary ordering is worse than sequentially processing one dataset after the other except when $c = 1/\lambda$. If the learning rate is small, then SGD should use the clean data first to aggressively proceed towards the optimum. If the learning rate is larger, then SGD should reserve the clean data for refining the initial estimates given by processing the noisy data.

4 ADAPTING THE LEARNING RATES TO THE NOISE LEVEL

We now investigate whether the performance of SGD can be improved by using different learning rates for oracles with different noise levels. Suppose we have oracles $\mathcal{G}_1$ and $\mathcal{G}_2$ with noise levels Γ_1 and Γ_2 that are implemented based on two datasets D_1 and D_2. Unlike the previous section, we do not assume any relation between Γ_1 and Γ_2 – we analyze the error for using oracle $\mathcal{G}_1$ followed by $\mathcal{G}_2$ in terms of Γ_1 and Γ_2 to choose a data order. Let $T = |D_1| + |D_2|$. Let $\beta_1 = \frac{|D_1|}{T}$ and $\beta_2 = 1 - \beta_1 = \frac{|D_2|}{T}$ be the fraction of the data coming from $\mathcal{G}_1$ and $\mathcal{G}_2$, respectively. We adapt the gradient updates in (4) to heterogeneous noise by choosing the learning rate η_t as a function of the noise level. Algorithm 1 shows a modified SGD for heterogeneous learning rates.

Algorithm 1 SGD with varying learning rate

```
1: Inputs: Oracles $\mathcal{G}_1, \mathcal{G}_2$ implemented by data sets $D_1, D_2$. Learning rates $c_1$ and $c_2$.
2: Set $w_1 = 0$.
3: for $t = 1, 2, \ldots, |D_1|$ do
4:     $w_{t+1} = \Pi_{\mathcal{W}}\left(w_t - \frac{c_1}{t}\mathcal{G}_1(w_t)\right)$
5: end for
6: for $t = |D_1|+1, |D_1|+2, \ldots, |D_1|+|D_2|$ do
7:     $w_{t+1} = \Pi_{\mathcal{W}}\left(w_t - \frac{c_2}{t}\mathcal{G}_2(w_t)\right)$
8: end for
9: return $w_{|D_1|+|D_2|+1}$.
```

Consider SGD with learning rate $\eta_t = c_1/t$ while querying $\mathcal{G}_1$ and with $\eta_t = c_2/t$ while querying $\mathcal{G}_2$ in the update (4). We must choose an order in which to query $\mathcal{G}_1$ and $\mathcal{G}_2$ as well as the constants c_1 and c_2 to get the best performance. We do this by minimizing an upper bound on the distance between the final iterate w_{T+1} and the optimal solution w^* to $\mathbb{E}[f(w)]$ where f is defined in (1), and the expectation is with respect to the data distribution and the gradient noise; the upper bound we choose is based on Rakhlin et al. (2012). Note that for smooth functions f, a bound on the distance $\|w_{T+1} - w^*\|$ automatically translates to a bound on the regret $f(w_{T+1}) - f(w^*)$.

Theorem 3 generalizes the results of Rakhlin et al. (2012) to our heterogeneous noise setting; the proof is in the supplement.

Theorem 3. *If $2\lambda c_1 > 1$ and if $2\lambda c_2 \neq 1$, and if we query $\mathcal{G}_1$ before $\mathcal{G}_2$ with learning rates c_1/t and c_2/t respectively, then the SGD algorithm satisfies*

$$\mathbb{E}\left[\|w_{T+1} - w^*\|^2\right] \leq \frac{4\Gamma_1^2}{T} \cdot \frac{\beta_1^{2\lambda c_2 - 1} c_1^2}{2\lambda c_1 - 1} + \frac{4\Gamma_2^2}{T} \cdot \frac{(1 - \beta_1^{2\lambda c_2 - 1}) c_2^2}{2\lambda c_2 - 1} + \mathcal{O}\left(\frac{1}{T^{\min(2, 2\lambda c_1)}}\right). \quad (9)$$

Two remarks are in order. First, the first two terms in the right hand side dominate the other term. Second, our proof techniques for Theorem 3, adapted from Rakhlin et al. (2012), require that $2\lambda c_1 > 1$ in order to get a $O(1/T)$ rate of convergence; without this condition, the dependence on T is $\Omega(1/T)$.

4.1 Algorithm description

Our algorithm for selecting c_1 and c_2 is motivated by Theorem 3. We propose an algorithm that selects c_1

and c_2 by minimizing the quantity $B(c_1, c_2)$ which represents the highest order terms in Theorem 3:

$$B(c_1, c_2) = \frac{4\Gamma_1^2 \beta_1^{2\lambda c_2 - 1} c_1^2}{T(2\lambda c_1 - 1)} + \frac{4\Gamma_2^2 (1 - \beta_1^{2\lambda c_2 - 1}) c_2^2}{T(2\lambda c_2 - 1)}. \tag{10}$$

Given λ, Γ_1, Γ_2 and β_1, we use c_1^* and c_2^* to denote the values of c_1 and c_2 that minimize $B(c_1, c_2)$. We can optimize for fixed c_2 with respect to c_1 by minimizing $\frac{c_1^2}{2\lambda c_1 - 1}$; this gives $c_1^* = 1/\lambda$, and $\frac{c_1^{*2}}{2\lambda c_1^* - 1} = 1/\lambda^2$, which is independent of β_1 or the noise levels Γ_1 and Γ_2. Minimizing $B(c_1^*, c_2)$ with respect to c_2 can be now performed numerically to yield $c_2^* = \mathbf{argmin}_{c_2} B(c_1^*, c_2)$. This yields optimal values of c_1 and c_2.

Now suppose we have two oracles $\mathcal{G}_\mathsf{C}$, $\mathcal{G}_\mathsf{N}$ with noise levels Γ_C and Γ_N that are implemented based on datasets D_C and D_N respectively. Let $\Gamma_\mathsf{C} < \Gamma_\mathsf{N}$, and let $\beta_\mathsf{C} = \frac{|D_\mathsf{C}|}{|D_\mathsf{C}| + |D_\mathsf{N}|}$ and $\beta_\mathsf{N} = \frac{|D_\mathsf{N}|}{|D_\mathsf{C}| + |D_\mathsf{N}|}$ be the fraction of the total data in each data set. Define the following functions:

$$H_\mathsf{CN}(c) = \frac{4\Gamma_\mathsf{C}^2 \beta_\mathsf{C}^{2\lambda c - 1}}{\lambda^2} + \frac{4\Gamma_\mathsf{N}^2 (1 - \beta_\mathsf{C}^{2\lambda c - 1}) c^2}{2\lambda c - 1},$$
$$H_\mathsf{NC}(c) = \frac{4\Gamma_\mathsf{N}^2 \beta_\mathsf{N}^{2\lambda c - 1}}{\lambda^2} + \frac{4\Gamma_\mathsf{C}^2 (1 - \beta_\mathsf{N}^{2\lambda c - 1}) c^2}{2\lambda c - 1}.$$

These represent the constant of the leading term in the upper bound in Theorem 3 for $(\mathcal{G}_1, \mathcal{G}_2) = (\mathcal{G}_\mathsf{C}, \mathcal{G}_\mathsf{N})$ and $(\mathcal{G}_1, \mathcal{G}_2) = (\mathcal{G}_\mathsf{N}, \mathcal{G}_\mathsf{C})$, respectively.
Algorithm 2 repeats the process of choosing optimal c_1, c_2 with two orderings of the data – $\mathcal{G}_\mathsf{C}$ first and $\mathcal{G}_\mathsf{N}$ first – and selects the solution which provides the best bounds (according to the higher order terms of Theorem 3).

Algorithm 2 Selecting the Learning Rates

```
1: Inputs: Data sets $D_C$ and $D_N$ accessed through
   oracles $\mathcal{G}_C$ and $\mathcal{G}_N$ with noise levels $\Gamma_C$ and $\Gamma_N$.
2: Let $\beta_C = \frac{|D_C|}{|D_C|+|D_N|}$ and $\beta_N = \frac{|D_N|}{|D_C|+|D_N|}$.
3: Calculate $c_{CN} = \mathbf{argmin}_c H_{CN}(c)$ and $c_{NC} = \mathbf{argmin}_c H_{NC}(c)$.
4: if $H_{CN}(c_{CN}) \le H_{NC}(c_{NC})$ then
5:     Run Algorithm 1 using oracles $(\mathcal{G}_C, \mathcal{G}_N)$, learning
       rates $c_1 = \frac{1}{\lambda}$ and $c_2 = c_{CN}$.
6: else
7:     Run Algorithm 1 using oracles $(\mathcal{G}_N, \mathcal{G}_C)$, learning
       rates $c_1 = \frac{1}{\lambda}$ and $c_2 = c_{NC}$.
8: end if
```

4.2 Regret Bounds

To provide a regret bound on the performance of SGD with two learning rates, we need to plug the optimal values of c_1 and c_2 into the right hand side of (10). Observe that as $c_1 = c_2$ and $c_2 = 0$ are feasible inputs to (10), our algorithm by construction has a superior regret bound than using a single learning rate only, or using clean data only.

Unfortunately, the value of c_2 that minimizes (10) does not have a closed form solution, and as such it is difficult to provide a general simplified regret bound that holds for all Γ_1, Γ_2 and β_1. In this section, we consider two cases of interest, and derive simplified versions of the regret bound for SGD with two learning rates for these cases.

We consider the two data orders $(\Gamma_1, \Gamma_2) = (\Gamma_\mathsf{N}, \Gamma_\mathsf{C})$ and $(\Gamma_1, \Gamma_2) = (\Gamma_\mathsf{C}, \Gamma_\mathsf{N})$ in a scenario where $\Gamma_\mathsf{N}/\Gamma_\mathsf{C} \gg 1$ and both β_N and β_C are bounded away from 0 and 1. That is, the noisy data is much noisier. The following two lemmas provide upper and lower bounds on $B(c_1^*, c_2^*)$ in this setting.

Lemma 1. *Suppose* $(\Gamma_1, \Gamma_2) = (\Gamma_\mathsf{N}, \Gamma_\mathsf{C})$ *and* $0 < \beta_\mathsf{N} < 1$. *Then for sufficiently large* $\Gamma_\mathsf{N}/\Gamma_\mathsf{C}$*, the optimal solution* c_2^* *to* (10) *satisfies*

$$2c_2^*\lambda \in \left[1 + \frac{2\log(\Gamma_\mathsf{N}/\Gamma_\mathsf{C}) + \log\log(1/\beta_\mathsf{N})}{\log(1/\beta_\mathsf{N})}, \right.$$
$$\left. 1 + \frac{2\log(4\Gamma_\mathsf{N}/\Gamma_\mathsf{C}) + \log\log(1/\beta_\mathsf{N})}{\log(1/\beta_\mathsf{N})}\right].$$

Moreover, $B(c_1^*, c_2^*)$ *satisfies:*

$$B(c_1^*, c_2^*) \ge \frac{4\Gamma_\mathsf{C}^2 (\log(\frac{\Gamma_\mathsf{N}}{\Gamma_\mathsf{C}}) + \frac{1}{2}\log\log\frac{1}{\beta_\mathsf{N}})}{\lambda^2 T \log(\frac{1}{\beta_\mathsf{N}})}$$
$$B(c_1^*, c_2^*) \le \frac{4\Gamma_\mathsf{C}^2}{\lambda^2 T}\left(4 + \frac{4 + 2\log(\frac{\Gamma_\mathsf{N}}{\Gamma_\mathsf{C}}) + \log\log(\frac{1}{\beta_\mathsf{N}})}{\log(\frac{1}{\beta_\mathsf{N}})}\right).$$

Observe that the regret bound grows logarithmically with $\Gamma_\mathsf{N}/\Gamma_\mathsf{C}$. Moreover, if we only used the cleaner data, then the regret bound would be $\frac{4\Gamma_\mathsf{C}^2}{\lambda^2 \beta_\mathsf{C} T}$, which is better, especially for large $\Gamma_\mathsf{N}/\Gamma_\mathsf{C}$. This means that using two learning rates with the noisy data first gives poor results at high noise levels.

Our second bound takes the opposite data order, processing the clean data first.

Lemma 2. *Suppose* $(\Gamma_1, \Gamma_2) = (\Gamma_\mathsf{C}, \Gamma_\mathsf{N})$ *and* $0 < \beta_\mathsf{C} < 1$. *Let* $\sigma = (\Gamma_\mathsf{N}/\Gamma_\mathsf{C})^{-2}$. *Then for sufficiently large* $\Gamma_\mathsf{N}/\Gamma_\mathsf{C}$*, the optimal solution* c_2^* *to* (10) *satisfies:* $2c_2^*\lambda \in \left[\sigma, \frac{8}{\beta_\mathsf{C}}\sigma\right]$. *Moreover,* $B(c_1^*, c_2^*)$ *satisfies:*

$$B(c_1^*, c_2^*) \ge \frac{4\Gamma_\mathsf{C}^2}{\lambda^2 \beta_\mathsf{C} T} \beta_\mathsf{C}^{8\sigma/\beta_\mathsf{C}}$$
$$B(c_1^*, c_2^*) \le \frac{4\Gamma_\mathsf{C}^2}{\lambda^2 \beta_\mathsf{C} T} \beta_\mathsf{C}^{\sigma}\left(1 + \sigma\frac{\log(1/\beta_\mathsf{C})}{4}\right).$$

If we only used the clean dataset, then the regret bound would be $\frac{4\Gamma_{\mathsf{C}}^2}{\lambda^2 \beta_{\mathsf{C}} T}$, so Lemma 2 yields an improvement by a factor of $\beta_{\mathsf{C}}^{(\Gamma_{\mathsf{N}}/\Gamma_{\mathsf{C}})^{-2}} \left(1 + \left(\frac{\Gamma_{\mathsf{N}}}{\Gamma_{\mathsf{C}}}\right)^{-2} \frac{\log(1/\beta_{\mathsf{C}})}{4}\right)$. As $\beta_{\mathsf{C}} < 1$, observe that this factor is always less than 1, and tends to 1 as $\Gamma_{\mathsf{N}}/\Gamma_{\mathsf{C}}$ tends to infinity; therefore the difference between the regret bounds narrows as the noisy data grows noisier. We conclude that using two learning rates with clean data first gives a better regret bound than using only clean data or using two learning rates with noisy data first.

5 EXPERIMENTS

We next illustrate our theoretical results through experiments on real data. We consider the task of training a regularized logistic regression classifier for binary classification under local differential privacy. For our experiments, we consider two real datasets – `MNIST` (with the task 1 vs. Rest) and `Covertype` (Type 2 vs. Rest). The former consists of 60,000 samples in 784 dimensions, while the latter consists of 500,000 samples in 54-dimensions. We reduce the dimension of the `MNIST` dataset to 25 via random projections.

To investigate the effect of heterogeneous noise, we divide the training data into subsets $(D_{\mathsf{C}}, D_{\mathsf{N}})$ to be accessed through oracles $(\mathcal{G}_{\mathsf{C}}, \mathcal{G}_{\mathsf{N}})$ with privacy parameters $(\epsilon_{\mathsf{C}}, \epsilon_{\mathsf{N}})$ respectively. We pick $\epsilon_{\mathsf{C}} > \epsilon_{\mathsf{N}}$, so $\mathcal{G}_{\mathsf{N}}$ is noisier than $\mathcal{G}_{\mathsf{C}}$. To simulate typical practical situations where cleaner data is rare, we set the size of D_{C} to be $\beta_{\mathsf{C}} = 10\%$ of the total data size. We set the regularization parameter $\lambda = 10^{-3}$, Γ_C and Γ_N according to Theorem 1 and use SGD with mini-batching (batch size 50).

Does Data Order Change Performance? Our first task is to investigate the effect of data order on performance. For this purpose, we compare three methods – CleanFirst, where all of D_{C} is used before D_{N}, NoisyFirst, where all of D_{N} is used before D_{C}, and Arbitrary, where data from $D_{\mathsf{N}} \cup D_{\mathsf{C}}$ is presented to the algorithm in a random order.

The results are in Figures 1(a) and 1(d). We use $\epsilon_C = 10, \epsilon_N = 3$. For each algorithm, we plot $|f(w_{T+1}) - f(v_{T+1})|$ as a function of the constant c in the learning rate. Here $f(w_{T+1})$ is the function value obtained after T rounds of SGD, and $f(v_{T+1})$ is the function value obtained after T rounds of SGD if we iterate over the data in the same order, but add no extra noise to the gradient. (See Theorem 2 for more details.) As predicted by Theorem 2, the results show that for $c < \frac{1}{\lambda}$, CleanFirst has the best performance, while for $c > \frac{1}{\lambda}$, NoisyFirst performs best. Arbitrary performs close to NoisyFirst for a range of values of c, which we expect as only 10% of the data belongs to D_{C}.

Are Two Learning Rates Better than One? We next investigate whether using two learning rates in SGD can improve performance. We compare five approaches. Optimal is the gold standard where we access the raw data without any intervening noisy oracle. CleanOnly uses only D_{C} with learning rate with the optimal value of c obtained from Section 4. SameClean and SameNoisy use a single value of the constant c in the learning rate for $D_{\mathsf{N}} \cup D_{\mathsf{C}}$, where c is obtained by optimizing (10)[1] under the constraint that $c_1 = c_2$. SameClean uses all of D_{C} before using D_{N}, while SameNoisy uses all of D_{N} before using D_{C}. In Algorithm2, we use Algorithm 2 to set the two learning rates and the data order (D_{C} first or D_{N} first). In each case, we set $\epsilon_{\mathsf{C}} = 10$, vary ϵ_{N} from 1 to 10, and plot the function value obtained at the end of the optimization.

The results are plotted in Figures 1(b) and 1(e). Each plotted point is an average of 100 runs. It is clear that Algorithm2, which uses two learning rates, performs better than both SameNoisy and SameClean. As expected, the performance difference diminishes as ϵ_{N} increases (that is, the noisy data gets cleaner). For moderate and high ϵ_{N}, Algorithm2 performs best, while for low ϵ_{N} (very noisy D_{N}), CleanOnly has slightly better performance. We therefore conclude that using two learning rates is better than using a single learning rate with both datasets, and that Algorithm2 performs best for moderate to low noise levels.

Does Noisy Data Always Help? A natural question to ask is whether using noisy data always helps performance, or if there is some threshold noise level beyond which we should not use noisy data. Lemma 2 shows that in theory, we obtain a better upper bound on performance when we use noisy data; in contrast, Figures 1(b) and 1(e) show that for low ϵ_{N} (high noise), Algorithm2 performs worse than CleanOnly. How do we explain this apparent contradiction?

To understand this effect, in Figures 1(c) and 1(f) we plot the performance of SGD using two learning rates (with $c_1 = \frac{1}{\lambda}$) against CleanOnly as a function of the second learning rate c_2. The figures show that the best performance is attained at a value of c_2 which is different from the value predicted by Algorithm2, and *this best performance is better than* CleanOnly. Thus, noisy data always improves performance; however, the improvement may not be achieved at the learning rate predicted by our algorithm.

[1]Note that we plug in separate noise rates for $\mathcal{G}_{\mathsf{C}}$ and $\mathcal{G}_{\mathsf{N}}$ in the learning rate calculations.

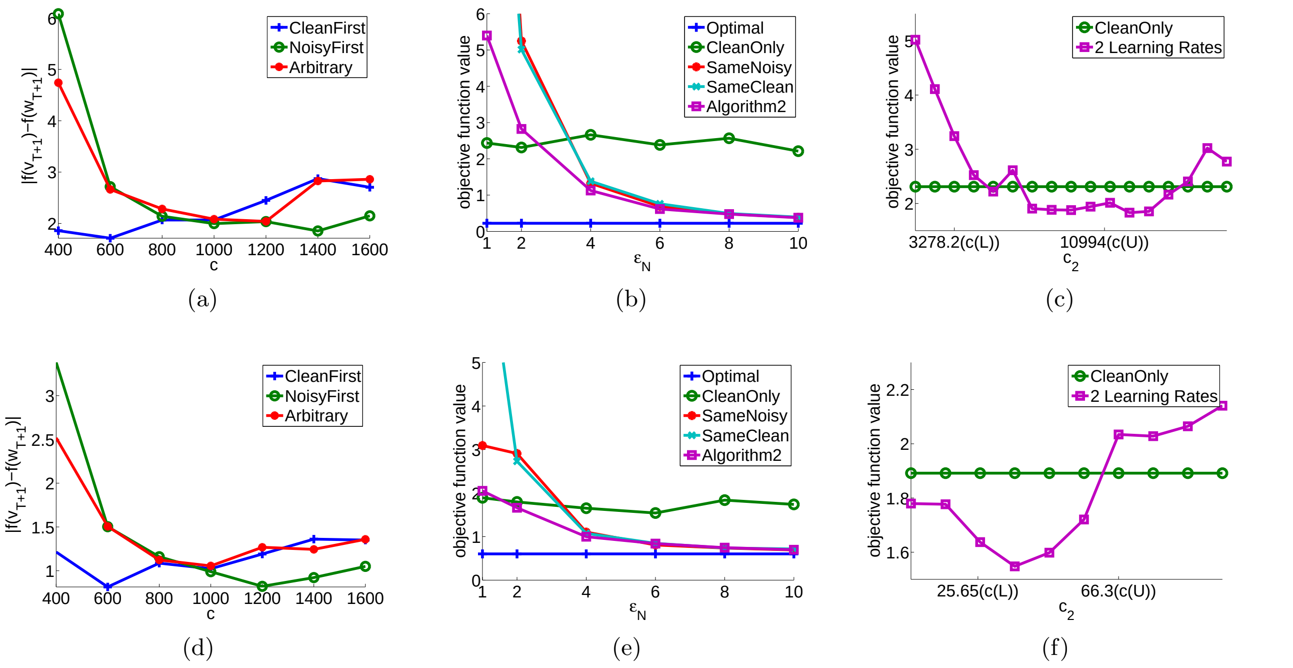

Figure 1: Column 1 plots $|f(w_{T+1}) - f(v_{T+1})|$ vs. constant c for $\lambda = 0.001$. Column 2 plots final objective function value vs. ϵ_N for $\epsilon_C = 10$. Column 3 plots final objective function value vs. c_2 for $\epsilon_N = 2$ (top) and $\epsilon_N = 1$ (bottom). Top row shows figures for `MNIST` and bottom row for `Covertype`.

Why does our algorithm perform suboptimally? We believe this happens because the values of Γ_N and Γ_C used by our algorithm are fairly loose upper bounds. For local differential privacy, an easy lower bound on Γ is $\sqrt{\frac{4(d^2+d)}{\epsilon^2 b}}$, where b is the mini-batch size; let $c_2(L)$ (resp. $c_2(U)$) be the value of c_2 obtained by plugging in these lower bounds (resp. upper bounds from Theorem 1) to Algorithm 1. Our experiments show that the optimal value of c_2 always lies between $c_2(L)$ and $c_2(U)$, which indicates that the suboptimal performance may be due to the looseness in the bounds.

We thus find that even in these high noise cases, theoretical analysis often allows us to identify *an interval* containing the optimal value of c_2. In practice, we recommend running Algorithm 2 twice – once with upper, and once with lower bounds to obtain an interval containing c_2, and then performing a line search to find the optimal c_2.

6 CONCLUSION

In this paper we propose a model for learning from heterogeneous noise that is appropriate for studying stochastic gradient approaches to learning. In our model, data from different sites are accessed through different oracles which provide noisy versions of the gradient. Learning under local differential privacy and random classification noise are both instances of our model. We show that for two sites with different noise levels, processing data from one site followed by the other is better than randomly sampling the data, and the optimal data order depends on the learning rate. We then provide a method for choosing learning rates that depends on the noise levels and showed that these choices achieve lower regret than using a common learning rate. We validate these findings through experiments on two standard data sets and show that our method for choosing learning rates often yields improvements when the noise levels are moderate. In the case where one data set is much noisier than the other, we provide a different heuristic to choose a learning rate that improves the regret.

There are several different directions towards generalizing the work here. Firstly, extending the results to multiple sites and multiple noise levels will give more insights as to how to leverage large numbers of data sources. This leads naturally to cost and budgeting questions: how much should we pay for additional noisy data? Our results for data order do not depend on the actual noise levels, but rather their relative level. However, we use the noise levels to tune the learning rates for different sites. If bounds on the noise levels are available, we can still apply our heuristic. Adaptive approaches for estimating the noise levels while learning are also an interesting approach for future study.

References

Alekh Agarwal, Peter L. Bartlett, Pradeep Ravikumar, and Martin J. Wainwright. Information-theoretic lower bounds on the oracle convexity of convex optimization. In *Adv. NIPS 22*. MIT Press, 2009.

F. Bach and E. Moulines. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Adv. NIPS 24*. MIT Press, 2011.

R. Bassily, A. Thakurta, and A. Smith. Private empirical risk minimization, revisited. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, Philadelphia, PA, USA, October 2014.

Ron Bekkerman, Mikhail Bilenko, and John Langford. *Scaling up Machine Learning, Parallel and Distributed Approaches.* Cambridge University Press, 2011.

Leon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proc. 19th Int'l Conf. on Comp. Stat.*, pages 177–187, 2010.

K. Chaudhuri, C. Monteleoni, and A. D. Sarwate. Differentially private empirical risk minimization. *JMLR*, 12:1069–1109, March 2011.

K Crammer, M Kearns, and J Wortman. Learning from data of variable quality. In Y. Weiss, B. Schölkopf, and J.C. Platt., editors, *Advances in Neural Information Processing Systems 18*, pages 219–226. MIT Press, 2006.

J. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *JMLR*, 2009.

John Duchi, Michael Jordan, and Martin Wainwright. Privacy aware learning. In P. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1439–1447. MIT Press, 2012.

John Duchi, Martin J. Wainwright, and Michael Jordan. Local privacy and minimax bounds: Sharp rates for probability estimation. In *Advances in Neural Information Processing Systems 26*, pages 1529–1537, 2013.

C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 486–503, Berlin, Heidelberg, 2006a. Springer-Verlag. doi: 10.1007/11761679_29. URL `http://dx.doi.org/10.1007/11761679_29`.

C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, 2006b.

P. Jain, P. Kothari, and A. Thakurta. Differentially private online learning. *JMLR – COLT*, 2012.

S. A. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? In *FOCS*, 2008.

Michael Kearns. Efficient noise-tolerant learning from statistical queries. *JACM*, 1998.

D. Kifer, A. Smith, and A. Thakurta. Private convex optimization for empirical risk minimization with applications to high-dimensional regression. In *COLT*, 2012.

N Natarajan, I Dhillon, P Ravikumar, and A Tewari. Learning with noisy labels. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 1196–1204. Curran Associates, Inc., 2013.

Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alex Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM J. Opt.*, 19(4):1574–1609, 2009.

A.S. Nemirovsky and D.B. Yudin. *Problem complexity and method efficiency in optimization.* Wiley & Sons, 1983.

Y. Nesterov and J. Vial. Confidence level solutions for stochastic programming. *Automatica*, 2008.

Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. Technical Report arXiv:1109.5647 [cs.LG], ArXiV, 2012.

B. Rubinstein, P. Bartlett, L. Huang, and N. Taft. Learning in a large function space: Privacy-preserving mechanisms for svm learning. *JPC*, 2013.

S Shalev-Shwartz, O Shamir, N Srebro, and K Sridaran. Stochastic convex optimization. In *COLT*, 2009.

Shuang Song, Kamalika Chaudhuri, and Anand Sarwate. Stochastic gradient descent with differentially private updates. In *GlobalSIP Conference*, 2013.

L. Wasserman and S. Zhou. A statistical framework for differential privacy. *JASA*, 2010.

Lin Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *JMLR*, 2010.