CS839: Differential Privacy and Learning March 03, 2026

Lecture 13: DP-GD for Generalized Linear Models

Instructor: Gavin Brown *Scribe:* Yiheng Su

Disclaimer: This document is intended as an informal supplement to in-class note-taking. It has not been given the level of scrutiny expected in polished lecture notes, let alone that reserved for peer-reviewed publications.

In this lecture, we continue our discussion of tools and theory for private optimization. We will (start to) analyze DP gradient descent for a broad class of optimization functions which arise from linear models. The key feature here is that our final excess loss bound will not explicitly depend on the underlying dimension of the problem.

In this class, our optimization focus is on variants of DP gradient descent. This is by far the most widely used method in practice. There are several other techniques for DP optimization, including

- the exponential mechanism
- "output perturbation"
- "objective perturbation"
- zero-th order optimization
- higher-order optimization

All of these have been explored in the literature to some extent, but they all have some drawbacks that result in DP gradient descent begin the primary tool. In particular, all of the above except zero-th order methods are ill-suited to the high-dimensional and nonconvex landscape induced by training neural networks.

1 Generalized Linear Models (GLMs)

There is a huge literature on generalized linear models, but we will not need it. Our results will really only use one property of GLMs, which we formalize in Claim 1.4.

Definition 1.1 (Generalized linear model)**.** Let

$$D = \{(x_1, y_1), \ldots, (x_n, y_n)\}, \qquad x_i \in \mathbb{R}^d, \qquad y_i \in \mathcal{Y},$$

where $\mathcal{Y}$ is the response space (for example, $\mathbb{R}$ or $\{0, 1\}$). For generalized linear models, each model is associated with a parameter vector $\theta \in \mathbb{R}^d$ and the model's prediction $\hat{y}$ depends on x and θ only through the inner product $\langle x, \theta \rangle$. The loss takes the form

$$\ell(\theta; x_i, y_i) = \ell(\langle x_i, \theta \rangle; y_i)$$

for some function ℓ.

In some parts of the literature this is called a "generalized linear problem." In some presentations you will see a function $\phi : \mathbb{R}^d \to \mathbb{R}^p$ which serves as the *feature map*, in which case the GLM computes $\langle \phi(x), \theta \rangle$. This lecture we do not distinguish between the data and the features used for prediction, but the results apply to either setting.

Example 1.2 (Linear regression)**.** Here $y_i \in \mathbb{R}$ and the prediction is

$$\hat{y}_i = \langle x_i, \theta \rangle.$$

A standard loss is the squared loss

$$\ell(\theta; x_i, y_i) = (\langle x_i, \theta \rangle - y_i)^2.$$

Example 1.3 (Logistic regression)**.** Here $y_i \in \{0, 1\}$ and the prediction is

$$\hat{y}_i = \sigma(\langle x_i, \theta \rangle), \qquad \sigma(z) = \frac{1}{1 + e^{-z}}.$$

The usual loss is the cross-entropy loss

$$\ell(\theta; x_i, y_i) = -y_i \log(\hat{y}_i) - (1 - y_i) \log(1 - \hat{y}_i).$$

For simplicity, we assume throughout that the loss is differentiable.

Claim 1.4 (Gradient structure of a GLM)**.** *Suppose $\ell(\theta; x, y) = \ell(\langle x, \theta \rangle; y)$ and ℓ is differentiable in its first argument. Then there exists a scalar $a = a(\theta; x, y)$ such that*

$$\nabla_\theta \ell(\theta; x, y) = a\, x.$$

Proof. Let

$$z = \langle x, \theta \rangle.$$

By the chain rule, for each coordinate j,

$$\frac{\partial}{\partial \theta_j} \ell(\theta; x, y) = \frac{\partial \ell}{\partial z}(z; y) \cdot \frac{\partial z}{\partial \theta_j} = \frac{\partial \ell}{\partial z}(z; y)\, x_j.$$

Therefore,

$$\nabla_\theta \ell(\theta; x, y) = \frac{\partial \ell}{\partial z}(z; y)\, x.$$

So the gradient is always a scalar multiple of the feature vector x. □

2 Noisy Gradient Descent

We next consider a version of gradient descent which adds Gaussian noise to each gradient. This is sometimes called the *Langevin algorithm* or *unadjusted Langevin algorithm (ULA)*. With $\sigma = 0$ we recover standard full-batch gradient descent.

Exercise 2.1. Explain why, if we don't make any assumptions about ℓ, Algorithm 1 is *not* DP.

Algorithm 1 Noisy Gradient Descent

```
Require: Dataset $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, loss $\ell$, step size $\eta$, number of iterations $T$, noise
    scale $\sigma$
1: $\theta_0 \leftarrow 0$
2: for $t = 0, 1, \ldots, T-1$ do
3:     $g_t \leftarrow \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta \ell(\theta_t; x_i, y_i)$
4:     $\tilde{g}_t \leftarrow g_t + z_t$, where $z_t \sim \mathcal{N}(0, \sigma^2 I_d)$
5:     $\theta_{t+1} \leftarrow \theta_t - \eta \tilde{g}_t$
6: end for
7: return $\tilde{\theta} = \frac{1}{T}\sum_{t=0}^{T-1} \theta_t$
```

3 Privacy via Bounded Gradients

We omit the proof of privacy, but discuss how it might go. To show that Algorithm 1 is DP, we use three of our basic tools:

1. Each noisy gradient $\tilde{g}_t$ is DP (by the guarantees of the Gaussian mechansim)
2. Since we use T noisy gradients, composition[1] tells us the overall set of noisy gradients is DP.
3. Postprocessing: we obtain the parameter updates and final parameter only by looking at the noisy gradients (and not touching the data).

For the first step, to invoke the DP guarantees of the Gaussian mechanism we must control how much the empirical gradient can change when one data point in the dataset is modified. In other words, we need a bound on the global sensitivity of $\frac{1}{n}\sum_{i=1}^{n} \nabla_\theta \ell(\theta; x_i, y_i)$. This will come from a bound on the maximum ℓ_2 norm of the gradient.

There are three standard ways to obtain such a bound.

1. Gradient clipping: we can take each each per-example gradient g_{it} and replace it with $\bar{g}_{it} = \mathrm{CLIP}_L(g_{it}) = \min\{1, L/\|g_{it}\|_2\} g_{it}$. This forces each gradient to have ℓ_2 norm at most L. It's what is usually done for DP deep learning.
2. Data/parameter clipping: for many specific applications, we can bound the gradient norm by forcing the data and parameters to be bounded. For example, for linear regression, the gradient has the form $x_i(y_i - \langle x_i, \theta\rangle)$. So if we know that θ, x_i, and y_i are all not too big, then the gradient can't be too big, either.
3. By fiat: include in your theorem statement: "assume $\|\nabla_\theta \ell(\theta; x_i, y_i)\|_2 \le L$." This pushes the problem elsewhere. Because of this convenience, it's what we'll do today.

Accordingly, we make the following bounded-gradient assumption. Under this assumption, the empirical gradient has bounded sensitivity, so Gaussian noise with an appropriate scale is enough to make each iteration private. This yields the following privacy guarantee for Algorithm 1.

Assumption 3.1 (Bounded gradients)**.** For all θ, x, and y, $\|\nabla_\theta \ell(\theta; x, y)\|_2 \le L$.

[1]Note that we require *adaptive* composition, since the gradient query we make depends on the previous answers.

Claim 3.2. *If for all θ, x, y, $\|\nabla_\theta \ell(\theta; x, y)\|_2 \le L$, then Alg. 1 is (ϵ, δ)-DP for*

$$\sigma = \frac{2L\sqrt{T\log(1/\delta)}}{n\epsilon}.$$

4 A Rank-Based Utility Guarantee for GLMs

The GLM structure allows a stronger utility guarantee than the generic dimension-dependent bound.

Theorem 4.1 (Song, Steinke, Thakkar, and Thakurta [2021]). *Suppose $\ell(\theta; x, y) = \ell(\langle x, \theta\rangle, y)$ is convex in θ and always has bounded gradient $\|\nabla_\theta \ell(\theta; x, y)\|_2 \le L$. Let*

$$\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(\theta; x_i, y_i),$$

and let M be the projector onto the eigenspaces of $\sum_{i=1}^{n} x_i x_i^T$. For Alg. 1 with σ as in Claim 3.2, $T = n^2\epsilon^2$, and an appropriate learning rate, we have

$$\mathbb{E}[\mathcal{L}(\tilde{\theta})] - \mathcal{L}(\theta^*) \le \frac{L\|\theta^*\|_M \sqrt{1 + 2\,\mathrm{rank}(M)\log(1/\delta)}}{\epsilon n}.$$

To understand this theorem, we need to define the second moment matrix and the associated weighted norm.

Definition 4.2 (Second moment matrix)**.** Let

$$S := \sum_{i=1}^{n} x_i x_i^T \in \mathbb{R}^{d\times d}$$

be the second moment matrix of the feature vectors $x_1, \ldots, x_n$. Since S is symmetric positive semidefinite, it admits an eigendecomposition

$$S = U\Lambda U^T,$$

where U is orthonormal and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$ with $\lambda_i \ge 0$.

Define $a_i = \mathbb{1}\{\lambda_i > 0\}$, where $\mathbb{1}_{\{}\cdot\}$ denotes the indicator function which takes the value one when its argument is true and is zero otherwise. Then define

$$A = \mathrm{diag}(a_1, \ldots, a_d), \qquad a_i = \begin{cases} 1, & \lambda_i > 0, \\ 0, & \lambda_i = 0. \end{cases}$$

Then

$$M := UAU^T$$

is the orthogonal projector onto the span of $\{x_1, \ldots, x_n\}$, equivalently onto the eigenspaces of S corresponding to its nonzero eigenvalues.

The rank of M, then, is simply the count of eigenvalues of $\sum_i x_i x_i^T$ which are non-zero.

To interpret the theorem we also need the following definition.

Definition 4.3. For $v \in \mathbb{R}^d$, define $\|v\|_M = \sqrt{v^T M v}$.

This will be important in the proof, as it corresponds to measuring the length of vectors in the space defined by the data. It is a *semi-norm*: it satisfies the properties of a norm but may map non-zero vectors to zero.

To understand these definitions, we can work through the following exercise about what it means to be a "projector onto the eigenspace."

Claim 4.4. *Let S be a real, symmetric, posititive semi-definite matrix with eigendecomposition $U\Lambda U^T$, where U contains the orthonormal basis of eigenvectors $u_1, \ldots, u_d$ and $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_d)$ contains the eigenvalues. Since U is a basis, for any vector v there exist numbers $\{c_i\}_{i\in[d]}$ such that $v = \sum_i c_i u_i$. For M as defined above, we have $Mv = \sum_i a_i c_i u_i = \sum_i \mathbb{1}\{\lambda_i > 0\} c_i u_i$.*

Proof. Another way to write the eigendecomposition is as $S = \sum_i \lambda_i u_i u_i^T$. Likewise, $M = \sum_i a_i u_i u_i^T$. Then the proof uses the fact that the u_i's are orthogonal and normalized: for all i we have $u_i^T u_i = \|u_i\|_2^2 = 1$ and $u_i^T u_j = 0$ for all $i \neq j$. Writing out all the details, we have:

$$\begin{aligned} Mv &= \left(\sum_i a_i u_i u_i^T\right)\left(\sum_j c_i u_i\right) \\ &= \sum_i (a_i u_i u_i^T)(c_i u_i) + \sum_{i\neq j} (a_i u_i u_i^T)(c_j u_j) \\ &= \sum_i (a_i c_i)(u_i)(u_i^T u_i) + \sum_{i\neq j} (a_i c_i)(u_i)(u_i^T u_i) \\ &= \sum_i (a_i c_i)(u_i) \cdot 1 + \sum_{i\neq j} (a_i c_i) \cdot 0 \\ &= \sum_i (a_i c_i)(u_i). \end{aligned}$$

This is what we claimed. □

The proof of Theorem 4.1 will be given in the next lecture.

References

Shuang Song, Thomas Steinke, Om Thakkar, and Abhradeep Thakurta. Evading the curse of dimensionality in unconstrained private glms. In *International Conference on Artificial Intelligence and Statistics*, pages 2638–2646. PMLR, 2021.