Optimization for Machine Learning

December 16, 2024

Today: Lecture on stochastic gradient

Tomorrow (1.45pm - 5pm): Lab session!

Friday (1.45pm - 5pm?): Exercises + Course project

STOCHASTIC GRADIENT

Previously:

- Want to solve optimization problems that involve data

Example: Linear regression

$X \in \mathbb{R}^{m \times d}$ $X = \begin{bmatrix} x_1^T \\ \vdots \\ x_m^T \end{bmatrix}$ x_i^T: feature vector for sample i

$y \in \mathbb{R}^m$ $y = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}$ vector of labels

Goal: Find a linear model for the data, that is a vector $w \in \mathbb{R}^d$ such that $x_i^T w - y_i \approx 0$

Optimization problem: $\min_{w \in \mathbb{R}^d} \frac{1}{2m} \|Xw - y\|^2 = \frac{1}{2m} \sum_{i=1}^{m} (x_i^T w - y_i)^2$

↑ objective function

→ This problem is convex, thus

$$\left[w^* \in \operatorname*{argmin}_{w \in \mathbb{R}^d} \frac{1}{2m} \|Xw - y\|^2 \right] \Longleftrightarrow \left[\nabla f(w^*) = 0 \right]$$

where $f(w) = \frac{1}{2m} \|Xw - y\|^2$ is C^1 (i.e. ∇f exists!)

↑ set of solutions / global minima

→ We can solve the problem numerically using gradient descent

↓

Using an iterative method that converges towards the solution

Gradient descent

$$w_{k+1} = w_k - \alpha_k \nabla f(w_k)$$

"iterate at iteration k"

stepsize > 0

→ If $\nabla f(w_k) = 0$, then w_k is a solution

→ otherwise, $f(w_{k+1}) < f(w_k)$ when α_k is sufficiently small

Important guarantee of gradient descent: Convergence rate

* In the convex case, after $K \geq 1$ iterations, we can guarantee that

$$f(w_K) - \min_{w \in \mathbb{R}^d} f(w) \leq O\left(\frac{1}{K}\right)$$

$\min_{w \in \mathbb{R}^d} f(w)$: minimum value of f

$O(A) = CxA$ $C > 0$ constant that does not depend on A

NB: In practice, we look at $f(w_K)$ or $f(w_K) - \tilde{f}$ with $\tilde{f} \approx \min_{w \in \mathbb{R}^d} f(w)$ instead of $f(w_K) - \min_{w \in \mathbb{R}^d} f(w)$

* You can do better than $\frac{1}{K}$! With accelerated gradient, we can get a rate $O\left(\frac{1}{K^2}\right)$ instead of $O\left(\frac{1}{K}\right)$

(NB: only true for convex functions)

* In the nonconvex case (f is not convex), gradient descent satisfies

$$\min_{0 \leq k \leq K-1} \|\nabla f(w_k)\| \leq O\left(\frac{1}{\sqrt{K}}\right)$$

after $K \geq 1$ iterations

→ Gradient descent converges to a point with zero gradient

→ But since f is nonconvex, this point is not necessarily a solution!

(Could be a local minimum but not a global one, a local maximum, etc)

↳ Gradient descent does not use the fact that f is defined by an average over data points, which is typically the case in machine learning: (Linear regression: $f(w) = \frac{1}{2n}\sum_{i=1}^{n}(x_i^T w - y_i)^2$)

Moreover:

- If $n \gg 1$, computing f or ∇f is expensive because it involves looking at the entire data!
- If $n \gg 1$, the data is likely to be correlated and finding good values for w_k is possible without looking at the entire data

⇒ For these reasons, the method of choice in ML is not gradient descent but stochastic gradient

Stochastic gradient: Basic algorithm

Setup: • minimize $f(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w)$ over $w \in \mathbb{R}^d$

where every f_i depends on the ith sample from a dataset of size n

- $\forall i = 1..m$, we assume that f_i is C^1 (so ∇f_i exists!)

$$\Rightarrow \nabla f(w) = \frac{1}{m} \sum_{i=1}^{m} \nabla f_i(w)$$

$$\underline{GD}: \quad w_{k+1} = w_k - \alpha_k \nabla f(w_k) = w_k - \frac{\alpha_k}{m} \sum_{i=1}^{m} \nabla f_i(w_k)$$

Stochastic Gradient: At every iteration, use one ∇f_i

$$w_{k+1} = w_k - \alpha_k \nabla f_{i_k}(w_k)$$

where i_k is chosen at random in $\{1, \dots, m\}$

$\alpha_k > 0$

→ Stochastic gradient is a randomized method: running it twice does not give the same output

→ What can we prove about stochastic gradient?

Theorem: Suppose that f is convex

Suppose that the indices $\{i_k\}$ in stochastic gradient are drawn independently and such that

1) $\mathbb{E}_{i_k}\left[\nabla f_{i_k}(w_k)\right] = \nabla f(w_k) = \frac{1}{m} \sum_{i=1}^{m} \nabla f_i(w_k)$

Expected value over i_k

$\mathbb{E}_{i_k}[\nabla f_{i_k}(w_k)] = \sum_{i=1}^{m} P(i_k = i) \times \nabla f_i(w_k)$

2) $\mathbb{E}_{i_k}\left[\|\nabla f_{i_k}(w_k)\|^2\right] - \|\nabla f(w_k)\|^2 \leq \sigma^2$

$\sigma^2 > 0$ "Variance"

1) $\Leftrightarrow$ On average, using $\nabla f_{i_k}(w_k)$ in the iteration is the same as using the full gradient $\nabla f(w_k)$

1) Holds when i_k is drawn uniformly $\mathbb{P}(i_k = i) = \frac{1}{n} \quad \forall i$

(2) $\Leftrightarrow$ On average, the norm $\|\nabla f_{i_k}(w_k)\|$ is not too far from $\|\nabla f(w_k)\|$

Then, under these assumptions, after $K \geq 1$ iterations, stochastic gradient computes w_K such that

$$\mathbb{E}\left[f(w_K) - \min_{w \in \mathbb{R}^d} f(w) \right] \leq O\left(\frac{1}{K}\right) + O(\sigma^2)$$

Comparison between gradient descent and stochastic gradient

For gradient descent, get $f(w_K) - \min_{w \in \mathbb{R}^d} f(w) \leq O\left(\frac{1}{K}\right)$ after $K \geq 1$ iterations

For stochastic gradient, get $\mathbb{E}\left[f(w_K) - \min_{w \in \mathbb{R}^d} f(w) \right] \leq O\left(\frac{1}{K}\right) + O(\sigma^2)$ after $K \geq 1$ iterations

$\mathbb{E}$: Average value of $f(w_K) - \min_{w \in \mathbb{R}^d} f(w)$ over $i_0, \dots, i_{K-1}$

$\Rightarrow$ The guarantee for gradient descent stronger

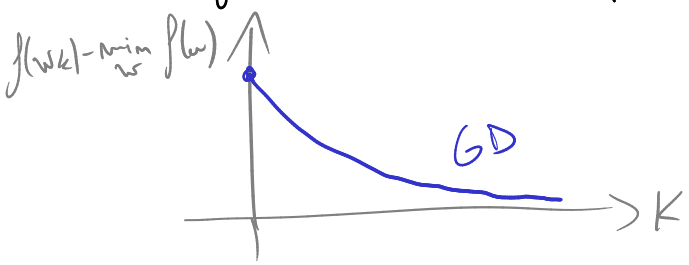
GD Gradient Descent

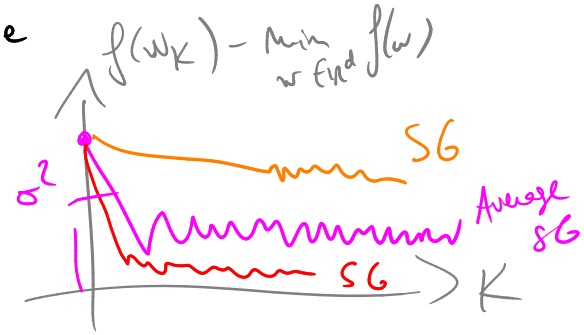
SG Stochastic Gradient

* it is deterministic while the guarantee for stochastic gradient is valid only on average
* the guarantee for GD implies $f(w_K) - \min_{w \in \mathbb{R}^d} f(w) \xrightarrow[K \to \infty]{} 0$

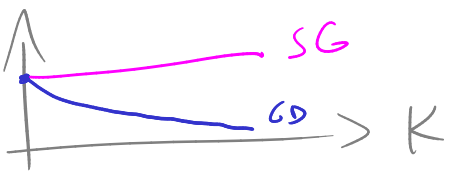
while the guarantee for SG implies $\mathbb{E}\left[f(w_K) - \min_{w \in \mathbb{R}^d} f(w) \right] \xrightarrow[K \to \infty]{} [0, \sigma^2]$

For SG, the function values converge to an interval that is a neighborhood of the optimal value





If we plot the curves together, we get



→ For a fixed number of iterations, gradient descent has better guarantees **BUT** an iteration of gradient descent does not have the same cost as an iteration of stochastic gradient

1 iteration of GD = 1 calculation of $\nabla f(w_k)$
= access to the entire dataset
= n accesses to a data point

1 iteration of SG = 1 calculation of $\nabla f_{i_k}(w_k)$
= 1 access to a data point

Definition: An epoch is a budget unit corresponding to n accesses to data points for a dataset of size n

→ With that unit of cost,

1 iteration of GD = 1 epoch
1 iteration of SG = $1/n$ epoch

Remark: SG: minimize $\frac{1}{m}\sum_{i=1}^{m} f_i(w)$ → Use 1 sample per iteration $i_t \in \{1, \dots, m\}$

dominant approach in ML

Sample Average Approximation: SAA: minimize $\frac{1}{m}\sum_{i=1}^{m} f_i(w)$ ⇒ minimize $\frac{1}{m}\sum_{i=1}^{m} f_i(w)$ $w \in \mathbb{R}^d$

alternative

$d \ll m$

→ For a fixed number of epochs N, we can run

- N iterations of gradient descent

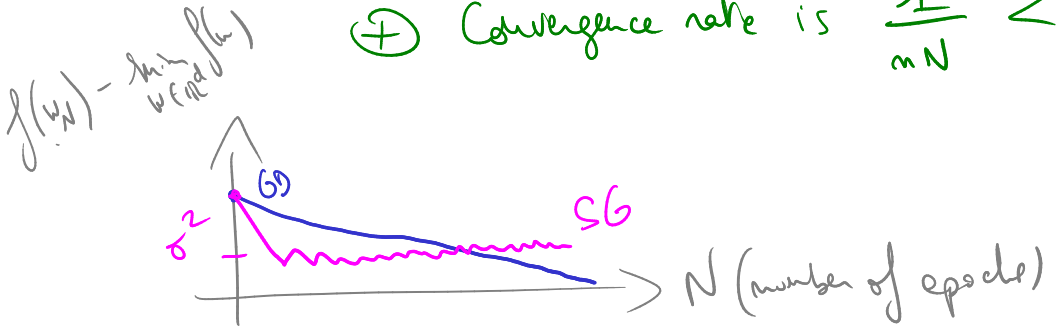
$$f(w_N) - \min_{w \in \mathbb{R}^d} f(w) \leq O\left(\frac{1}{N}\right)$$

- $m \times N$ iterations of stochastic gradient

$$\mathbb{E}\left[f(w_{mN}) - \min_{w \in \mathbb{R}^d} f(w)\right] \leq O\left(\frac{1}{mN}\right) + O(\sigma^2)$$

⊖ Convergence towards a neighborhood of the minimum value

⊕ Convergence rate is $\frac{1}{mN} \ll \frac{1}{N}$ when $m \gg 1$



Variants of stochastic gradient

1) Batch stochastic gradient

Basic SG: 1 sample / iteration GD: all samples / iteration

More general: Batch SG

Batch SG: $w_{k+1} = w_k - \alpha_k \times \frac{1}{|S_k|} \sum_{i \in S_k} \nabla f_i(w_k)$

$|S_k|$: number of elements of S_k

where S_k is a set of indices drawn randomly in $\{1, \dots, n\}$ with or without replacement

Special cases:
- $|S_k| = 1 \Rightarrow$ SG
- $S_k = \{1, \dots, n\}$ (n indices drawn without replacement) $\Rightarrow$ GD

→ Using a batch of samples is good when sample gradients (∇f_i) can be computed in parallel
⇒ that is why standard values for the batch size ($|S_k|$) are powers of 2, because they correspond to a number of cores or processors available

→ If $|S_k|$ is too close to n (large batch regime), the method behaves like GD

→ For $|S_k| \ll n$ and $|S_k| > 1$ (mini-batch regime), can get (but not always) better performance than stochastic gradient

⇒ Trade-off between better steps (large batch) and cheap iteration (small batch)

⇒ In the analysis, this appears in the convergence rate with $K \geq 1$ iterations of batch stochastic gradient with $|S_k| = m_b \in \{1, \dots, n\}$, get $\mathbb{E}[\dots] \leq O\left(\frac{m_b}{K}\right) + O\left(\frac{\sigma^2}{m_b}\right)$

true for indices drawn with replacement

2) Stochastic gradient methods for (mostly) deep learning

→ Most popular variants of SG in deep learning: Adagrad, **Adam**, SG with momentum, RMSProp, ...

Most important

→ Can all be written under the form

$$w_{k+1} = w_k - \alpha_k \, m_k \,/\, v_k$$

/ : componentwise division (numpy.divide)

$m_k \in \mathbb{R}^d$, vector of direction

$v_k \in \mathbb{R}^d$, vector of scaling for the stepsize

With $m_k = \nabla f_{i_k}(w_k)$ and $v_k = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$, recover SG

The "advanced" variants: use different m_k and v_k

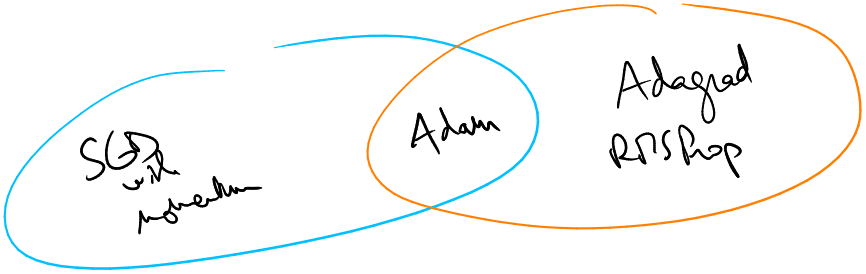
1) Momentum-based methods

Define m_k as a function of m_{k-1} and $\nabla f_{i_k}(w_k)$ (combine the new stochastic gradient with the previous step)

$\simeq$ Accelerated gradient

2) Scaling-based methods

Adjust the stepsize for each coordinate



Adam: 2015 method, most cited optimization paper (~198000 citations)