Lecture 8. Stochastic Gradient Descent

Bao Wang
Department of Mathematics
Scientific Computing and Imaging Institute
University of Utah
Math 5750/6880, Fall 2023

Loss function

- We have formulated training machine learning models as follows:

$$\min f(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_i(\boldsymbol{x}) + R(\boldsymbol{x})$$

where $\boldsymbol{x}$ is the parameter of the machine learning model, $\mathcal{L}_i(\boldsymbol{x})$ is the loss of the ith training instance, and $R(\boldsymbol{x})$ is the regularization term.

- How to find the optimal $\boldsymbol{x}^*$ if n is very large? Computing ∇f is very difficult.

- Let us first ignore the regularization term. (Using proximal!)

Empirical risk minimization

- Let $\{\boldsymbol{a}_i, y_i\}_{i=1}^n$ be n random samples, and consider

$$\min_{\boldsymbol{x}} F(x) := \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}; \{\boldsymbol{a}_i, y_i\}),$$

e.g. quadratic loss $f(\boldsymbol{x}; \{\boldsymbol{a}_i, y_i\}) = (\boldsymbol{a}_i^\top \boldsymbol{x} - y_i)^2$.

- If one draws index $j \sim Unif(1, \cdots, n)$ uniformly at random, then

$$F(\boldsymbol{x}) = \mathbb{E}_j[f(\boldsymbol{x}; \{\boldsymbol{a}_j, y_j\})].$$

- More generally, we consider the following stochastic programming

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x}; \xi)]$$

  > ξ: randomness in problem

  > suppose $f(\cdot, \xi)$ is convex for every ξ (and hence $F(\cdot)$ is convex)

A natural solution

- Under "mild" technical conditions, we have

$$\begin{aligned} \boldsymbol{x}^{t+1} &= \boldsymbol{x}^t - \eta_t \nabla F(\boldsymbol{x}^t) \\ &= \boldsymbol{x}^t - \eta_t \nabla \mathbb{E}[f(\boldsymbol{x}^t; \xi)] \\ &= \boldsymbol{x}^t - \eta_t \mathbb{E}[\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^t; \xi)] \end{aligned}$$

- Issues:
  - > distribution of ξ maybe unknown.
  - > even if it is known, evaluating high-dimensional expectations is often expensive.

Stochastic gradient descent (SGD)

- Stochastic approximation/stochastic gradient descent (SGD):

$$\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - \eta_t \boldsymbol{g}(\boldsymbol{x}^t; \xi^t) \tag{1}$$

where $\boldsymbol{g}(\boldsymbol{x}^t; \xi^t)$ is an unbiased estimate of $\nabla F(\boldsymbol{x}^t)$, i.e.

$$\mathbb{E}[\boldsymbol{g}(\boldsymbol{x}^t; \xi^t)] = \nabla F(\boldsymbol{x}^t).$$

- SGD is a stochastic algorithm for finding a critical point $\boldsymbol{x}$ obeying $\nabla F(\boldsymbol{x}) = 0$.

- **Example.** Consider the empirical risk minimization problem

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) := \frac{1}{n}\sum_{i=1}^{n} f(\boldsymbol{x}; \{\boldsymbol{a}_i, y_i\})$$

SGD for empirical risk minimization
for $t = 0, 1, \cdots$
choose i_t uniformly at random, and run

$$\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - \eta_t \nabla_{\boldsymbol{x}} f_{i_t}(\boldsymbol{x}^t; \{\boldsymbol{a}_i, y_i\})$$

- Sample with replacement. However, the most used SGD uses sample without replacement.

• **Benefits:** SGD exploits information more efficiently than batch methods (gradient descent)

1. practical data usually involve lots of redundancy; using all data simultaneously in each iteration might be inefficient.

2. SGD is particularly efficient at the very beginning, as it achieves fast initial improvement with very low per-iteration cost.

SGD for empirical risk minimization

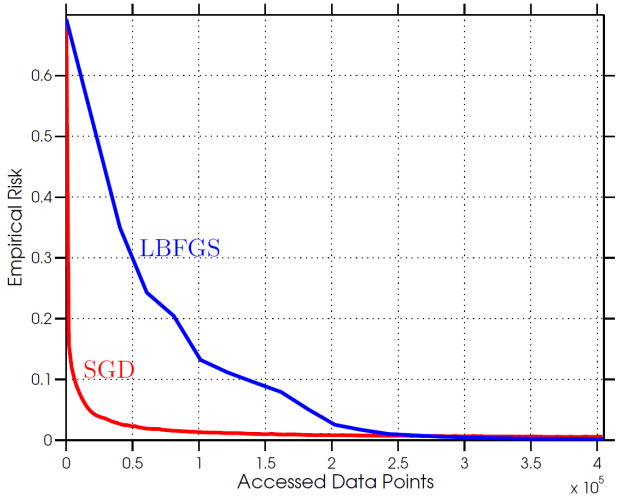

Figure: Logistic regression for RCV1 dataset ($\eta_t \equiv 4$).

Convergence analysis

Assumptions. Consider

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) := \mathbb{E}[f(\boldsymbol{x}; \xi)]$$

where

1. F: μ-strongly convex, L-smooth
2. $g(\boldsymbol{x}^t; \xi^t)$: an **unbiased** estimate of $\nabla F(\boldsymbol{x}^t)$ given $\{\xi^0, \cdots, \xi^{t-1}\}$
3. for all $\boldsymbol{x}$,

$$\mathbb{E}[\|\boldsymbol{g}(\boldsymbol{x}; \xi)\|_2^2] \leq \sigma_g^2 + c_g \|\nabla F(\boldsymbol{x})\|_2^2. \tag{2}$$

Bounded variance assumption and σ_g^2 measures the variance.

Theorem. [Convergence of SGD for strongly convex problems: fixed stepsizes] Under the assumptions above, if $\eta_t \equiv \eta \leq \frac{1}{Lc_g}$, then SGD (1) achieves

$$\mathbb{E}[F(\boldsymbol{x}^t) - F(\boldsymbol{x}^*)] \leq \frac{\eta L \sigma_g^2}{2\mu} + (1 - \eta\mu)^t (F(\boldsymbol{x}^0) - F(\boldsymbol{x}^*)).$$

Proof. See L. Bottou, F. Curtis, and J. Nocedal. "Optimization Methods for Large-Scale Machine Learning", SIAM Rev, 2018.

Remarks.

1. fast (linear) convergence at the very beginning
2. converges to some neighborhood of $\boldsymbol{x}^*$ — variance in gradient computation prevents further progress
3. when gradient computation is noiseless (i.e. $\sigma_g = 0$), it converges linearly to optimal points
4. smaller stepsizes η yield better converging points

One practical strategy

Run SGD with fixed stepsizes; whenever progress stalls, reduce stepsizes and continue SGD.

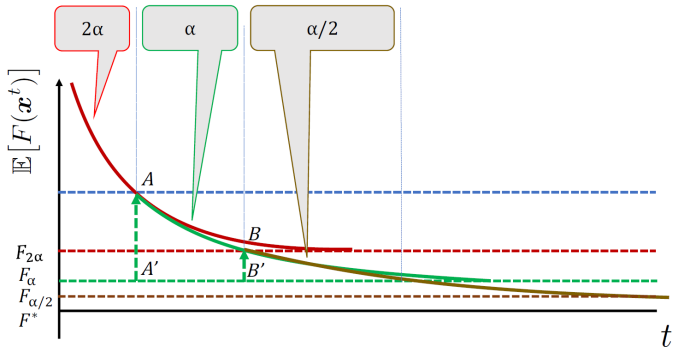

Figure: Whenever progress stalls, we half the stepsizes and repeat.

Stagewise stepsize.

Theorem. [Convergence of SGD for strongly convex problems: diminishing stepsizes] Suppose F is μ-strongly convex, and (2) holds with $c_g = 0$. If $\eta_t = \frac{\theta}{t+1}$ for some $\theta > \frac{1}{2\mu}$, then SGD (1) achieves

$$\mathbb{E}[\|\boldsymbol{x}^t - \boldsymbol{x}^*\|_2^2] \leq \frac{c_\theta}{t+1}$$

where $c_\theta = \max\left\{\frac{2\theta^2\sigma_g^2}{2\mu\theta - 1}, \|\boldsymbol{x}_0 - \boldsymbol{x}^*\|_2^2\right\}$.

Remark. convergence rate $O(1/t)$ with diminishing stepsize $\eta_t \sim 1/t$.

Proof. Using the SGD update rule, we have

$$\begin{aligned}\|\boldsymbol{x}^{t+1}-\boldsymbol{x}^*\|_2^2 &= \|\boldsymbol{x}^t-\eta_t\boldsymbol{g}(\boldsymbol{x}^t;\xi^t)-\boldsymbol{x}^*\|_2^2 \\ &= \|\boldsymbol{x}^t-\boldsymbol{x}^*\|_2^2-2\eta_t(\boldsymbol{x}^t-\boldsymbol{x}^*)^\top\boldsymbol{g}(\boldsymbol{x}^t;\xi^t)+\eta_t^2\|\boldsymbol{g}(\boldsymbol{x}^t;\xi^t)\|_2^2.\end{aligned} \tag{3}$$

Since $\boldsymbol{x}^t$ is independent of ξ_t, apply the law of total expectation to obtain

$$\begin{aligned}\mathbb{E}[(\boldsymbol{x}^t-\boldsymbol{x}^*)^\top\boldsymbol{g}(\boldsymbol{x}^t;\xi^t)] &= \mathbb{E}[\mathbb{E}[(\boldsymbol{x}^t-\boldsymbol{x}^*)^\top\boldsymbol{g}(\boldsymbol{x}^t;\xi^t)|\xi_1,\cdots,\xi_{t-1}] \\ &= \mathbb{E}[(\boldsymbol{x}^t-\boldsymbol{x}^*)^\top\mathbb{E}[\boldsymbol{g}(\boldsymbol{x}^t;\xi^t)|\xi_1,\cdots,\xi_{t-1}]] \\ &= \mathbb{E}[(\boldsymbol{x}^t-\boldsymbol{x}^*)^\top\nabla F(\boldsymbol{x}^t)]\end{aligned} \tag{4}$$

Furthermore, strong convexity gives

$$\begin{aligned}\langle \nabla F(\boldsymbol{x}^t), \boldsymbol{x}^t - \boldsymbol{x}^* \rangle &= \langle \nabla F(\boldsymbol{x}^t) - \underbrace{\nabla F(\boldsymbol{x}^*)}_{=0}, \boldsymbol{x}^t - \boldsymbol{x}^* \rangle \geq \mu \|\boldsymbol{x}^t - \boldsymbol{x}^*\|_2^2 \\ &\Rightarrow \mathbb{E}[\langle \nabla F(\boldsymbol{x}^t), \boldsymbol{x}^t - \boldsymbol{x}^* \rangle] \geq \mu \mathbb{E}[\|\boldsymbol{x}^t - \boldsymbol{x}^*\|_2^2].\end{aligned} \tag{5}$$

Combine (3), (4), (5) and (2) (with $c_g = 0$) to obtain

$$\mathbb{E}[\|\boldsymbol{x}^{t+1} - \boldsymbol{x}^*\|_2^2] \leq (1 - 2\mu\eta_t)\mathbb{E}[\|\boldsymbol{x}^t - \boldsymbol{x}^*\|_2^2] + \underbrace{\eta_t^2 \sigma_g^2}_{\text{does not vanish unless } \eta_t \to 0.} \tag{6}$$

Take $\eta_t = \frac{\theta}{t+1}$ and use induction to conclude the proof.

Optimality

Informally, when minimizing strongly convex functions, no algorithm performing t queries to noisy first-order oracles can achieves an accuracy better than the order of $1/t$

$\Rightarrow$ SGD with stepsizes $\eta_t \sim 1/t$ is optimal.

Optimality

[Nemirovski & Yudin 1983] More precisely, consider a class of problems in which f is μ-strongly convex and L-smooth, and $Var(\|\boldsymbol{g}(\boldsymbol{x}^t;\xi^t)\|_2) \leq \sigma^2$. Then the worst-case iteration complexity for (stochastic) first-order methods:

$$\sqrt{\frac{L}{\mu}} \log\left(\frac{L\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|_2^2}{\epsilon}\right) + \frac{\sigma^2}{\mu\epsilon}.$$

> For deterministic case: $\sigma = 0$, and hence the lower bound is

$$\sqrt{\frac{L}{\mu}} \log\left(\frac{L\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|_2^2}{\epsilon}\right) \quad \text{(achievable by Nesterov's method)}$$

> For noisy case with large σ, the lower bound is dominated by ($1/t$ convergence rate)

$$\frac{\sigma^2}{\mu} \cdot \frac{1}{\epsilon}$$

Nesterov's method for strongly convex problems:

$$\boldsymbol{x}^{t+1} = prox_{\eta_t h}(\boldsymbol{y}^t - \eta_t \nabla f(\boldsymbol{x}^t))$$
$$\boldsymbol{y}^{t+1} = \boldsymbol{x}^{t+1} + \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}(\boldsymbol{x}^{t+1} - \boldsymbol{x}^t)$$

Theorem 4. [Convergence of accelerated proximal gradient methods for strongly convex case] Suppose f is μ-strongly convex and L-smooth. If $\eta_t \equiv 1/L$, then

$$F(\boldsymbol{x}^t) - F^{opt} \leq \left(1 - \frac{1}{\sqrt{\kappa}}\right)^t \left(F(\boldsymbol{x}^0) - F^{opt} + \frac{\mu\|\boldsymbol{x}^0 - \boldsymbol{x}^*\|_2^2}{2}\right).$$

Comparison with batch GD

	iteration complexity	per-iteration cost	total comput. cost
batch GD	$\log \frac{1}{\varepsilon}$	n	$n \log \frac{1}{\varepsilon}$
SGD	$\frac{1}{\varepsilon}$	1	$\frac{1}{\varepsilon}$

Figure: Empirical risk minimization with n samples.

SGD is more appealing for large n and moderate accuracy ϵ (in which case $\frac{1}{\epsilon} < n \log \frac{1}{\epsilon}$) – which often arise in the big data regime! (n is very big.)

What if we lose strong convexity?

$$\min_{\boldsymbol{x}} F(\boldsymbol{x}) := \mathbb{E}[f(\boldsymbol{x}; \xi)]$$

1. F: convex
2. $\mathbb{E}[\|\boldsymbol{g}(\boldsymbol{x}; \xi)\|_2^2] \leq \sigma_g^2$ for all $\boldsymbol{x}$
3. $\boldsymbol{g}(\boldsymbol{x}^t; \xi^t)$ is an unbiased estimate of $\nabla F(\boldsymbol{x}^t)$ given $\{\xi^0, \cdots, \xi^{t-1}\}$.

Suppose we run SGD and return a weighted average

$$\tilde{\boldsymbol{x}}^t := \sum_{k=0}^{t} \frac{\eta_k}{\sum_{j=0}^{t} \eta_j} \boldsymbol{x}^k.$$

Theorem. Under the assumption above, one has

$$\mathbb{E}[F(\tilde{\boldsymbol{x}}^t) - F(\boldsymbol{x}^*)] \leq \frac{\frac{1}{2}\mathbb{E}[\|\boldsymbol{x}^0 - \boldsymbol{x}^*\|_2^2] + \frac{1}{2}\sigma_g^2 \sum_{k=0}^t \eta_k^2}{\sum_{k=0}^t \eta_k}.$$

In particular, if $\eta_t \sim 1/\sqrt{t}$, then

$$\mathbb{E}[F(\tilde{\boldsymbol{x}}^t) - F(\boldsymbol{x}^*)] \lesssim \frac{\log t}{\sqrt{t}}.$$

GD for cvx: $\eta_t = 1/L$ and convergence rate is $1/t$.

Proof. By convexity of F, we have

$$F(\boldsymbol{x}) \geq F(\boldsymbol{x}^t) + (\boldsymbol{x} - \boldsymbol{x}^t)^\top \nabla F(\boldsymbol{x}^t) \Rightarrow \mathbb{E}[(\boldsymbol{x}^t - \boldsymbol{x}^*)^\top \nabla F(\boldsymbol{x}^t)] \geq \mathbb{E}[F(\boldsymbol{x}^t) - F(\boldsymbol{x}^*)].$$

This together with (3) and (4) implies

$$2\eta_k \mathbb{E}[F(\boldsymbol{x}^k) - F(\boldsymbol{x}^*)] \leq \mathbb{E}[\|\boldsymbol{x}^k - \boldsymbol{x}^*\|_2^2] - \mathbb{E}[\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|_2^2] + \eta_k^2 \sigma_g^2.$$

Sum over $k = 0, \cdots, t$ to obtain

$$\begin{aligned}\sum_{k=0}^{t} 2\eta_k \mathbb{E}[F(\boldsymbol{x}^k) - F(\boldsymbol{x}^*)] &\leq \mathbb{E}[\|\boldsymbol{x}^0 - \boldsymbol{x}^*\|_2^2] - \mathbb{E}[\|\boldsymbol{x}^{t+1} - \boldsymbol{x}^*\|_2^2] + \sigma_g^2 \sum_{k=0}^{t} \eta_k^2 \\ &\leq \mathbb{E}[\|\boldsymbol{x}^0 - \boldsymbol{x}^*\|_2^2] + \sigma_g^2 \sum_{k=0}^{t} \eta_k^2.\end{aligned}$$

Setting $v_t = \frac{\eta_t}{\sum_{k=0}^{t} \eta_k}$ yields

$$\sum_{k=0}^{t} v_k \mathbb{E}[F(\boldsymbol{x}^k) - F(\boldsymbol{x}^*)] \leq \frac{\frac{1}{2}\mathbb{E}[\|\boldsymbol{x}^0 - \boldsymbol{x}^*\|_2^2] + \frac{1}{2}\sigma_g^2 \sum_{k=0}^{t} \eta_k^2}{\sum_{k=0}^{t} \eta_k}.$$

By convexity of F, we arrive at

$$\mathbb{E}[F(\tilde{\boldsymbol{x}}^t) - F(\boldsymbol{x}^*)] \leq \frac{\frac{1}{2}\mathbb{E}[\|\boldsymbol{x}^0 - \boldsymbol{x}^*\|_2^2] + \frac{1}{2}\sigma_g^2 \sum_{k=0}^{t} \eta_k^2}{\sum_{k=0}^{t} \eta_k}.$$

Reducing variance via iterate averaging

Stepsize choice $O(1/t)$?

Two conflicting regimes:

1. The noiseless case (i.e. $\boldsymbol{g}(\boldsymbol{x};\xi) = \nabla F(\boldsymbol{x})$): stepsizes $\eta_t \sim 1/t$ are way too conservative.

2. The general noisy case: longer stepsizes ($\eta_t \gg 1/t$) might fail to suppress noise (and hence slow down convergence).

Can we modify SGD so as to allow for larger stepsizes without compromising convergence rates?

Motivation for iteration averaging. SGD with long stepsizes poorly suppresses noise, which tends to oscillate around the global minimizers due to the noisy nature of gradient computation. One may, however, average iterates to mitigate oscillation and reduce variance.

Run SGD and return

$$\bar{\boldsymbol{x}}^t := \frac{1}{t}\sum_{i=0}^{t-1} \boldsymbol{x}^i, \tag{7}$$

with larger stepsizes $\eta_t \sim t^{-\alpha}$ with $\alpha < 1$.

Key idea: average the iterates to reduce variance and improve convergence.

Example: a toy quadratic problem

Consider $\min_{\boldsymbol{x}\in\mathbb{R}^d} \frac{1}{2}\|\boldsymbol{x}\|_2^2$ using constant stepsizes $\eta_t \equiv \eta < 1$. Assume the unbiased stochastic gradient satisfies

$$\boldsymbol{g}(\boldsymbol{x}^t;\xi^t) = \boldsymbol{x}^t + \xi^t \quad \text{with} \quad \mathbb{E}[\xi^t|\xi^0,\cdots,\xi^{i-1}] = 0 \quad \text{and} \quad \mathbb{E}[\xi^t\xi^{t\top}|\xi^0,\cdots,\xi^{t-1}] = \mathsf{I}.$$

SGD iterates:

$$\begin{aligned}
\boldsymbol{x}^1 &= \boldsymbol{x}^0 - \eta(\boldsymbol{x}^0 + \xi^0) = (1-\eta)\boldsymbol{x}^0 - \eta\xi^0 \\
\boldsymbol{x}^2 &= \boldsymbol{x}^1 - \eta(\boldsymbol{x}^1 + \xi^1) = (1-\eta)^2\boldsymbol{x}^0 - \eta(1-\eta)\xi^0 - \eta\xi^1 \\
&\vdots \\
\boldsymbol{x}^t &= (1-\eta)^t\boldsymbol{x}^0 - \eta(1-\eta)^{t-1}\xi^0 - \eta(1-\eta)^{t-2}\xi^1 - \cdots
\end{aligned}$$

Example: a toy quadratic problem

i.e.,

$$\bar{\boldsymbol{x}}^t \approx \underbrace{\frac{1}{t}\sum_{k=0}^{t-1}(1-\eta)^k \boldsymbol{x}^0}_{=\frac{1}{t}\frac{1-(1-\eta)^t}{\eta}\boldsymbol{x}^0 \to 0} - \underbrace{\eta\{1+(1-\eta)+\cdots\}\frac{1}{t}\sum_{k=0}^{t-1}\xi^k}_{\text{imprecise: but close enough for large t}}$$

$$\approx -\frac{1}{t}\sum_{k=0}^{t-1}\xi^k \quad (\text{since } \; 1+(1-\eta)+\cdots = \eta^{-1})$$

$$\to \frac{1}{\sqrt{t}}\mathcal{N}(0,\mathrm{I}) \text{ as } t\to\infty \quad (\text{the central limit theorem for martingale})$$

Other popular SGD variants: SGD with momentum

$$\begin{aligned} \boldsymbol{v}^{t+1} &= \gamma \boldsymbol{v}_t + \eta \boldsymbol{g}(\boldsymbol{x}^t; \xi^t) \\ \boldsymbol{x}^{t+1} &= \boldsymbol{x}^t - \boldsymbol{v}^{t+1}. \end{aligned} \tag{8}$$

How it is related to the heavy-ball method?

From the second equation, we have $\boldsymbol{v}^{t+1} = \boldsymbol{x}^t - \boldsymbol{x}^{t+1}$, and therefore we have

$$\boldsymbol{x}^t - \boldsymbol{x}^{t+1} = \gamma(\boldsymbol{x}^{t-1} - \boldsymbol{x}^t) + \eta \boldsymbol{g}(\boldsymbol{x}^t; \xi^t),$$

i.e.

$$\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - \eta \boldsymbol{g}(\boldsymbol{x}^t; \xi^t) + \gamma(\boldsymbol{x}^t - \boldsymbol{x}^{t-1}).$$

Other popular SGD variants: SGD with Nesterov momentum

$$\begin{aligned} \boldsymbol{v}^{t+1} &= \gamma \boldsymbol{v}_t + \eta \boldsymbol{g}(\boldsymbol{x}^t - \gamma \boldsymbol{v}_t; \xi^t) \\ \boldsymbol{x}^{t+1} &= \boldsymbol{x}^t - \boldsymbol{v}^{t+1}. \end{aligned} \tag{9}$$

In practice, SGD with Nesterov momentum performs similar to SGD with momentum.

Other popular SGD variants: Adagrad

Let $g_i^t = \boldsymbol{g}(x_i^t)$, SGD update for every parameter x_i at each time step t then becomes

$$x_i^{t+1} = x_i^t - \eta g_i^t.$$

Adagrad modifies the general learning rate η at each time step t for every parameter x_i based on the past gradients that have been computed for x_i:

$$x_i^{t+1} = x_i^t - \frac{\eta}{\sqrt{G_{ii}^t + \epsilon}} \cdot g_i^t,$$

where $G^t \in \mathbb{R}^{d \times d}$ is a diagonal matrix where each diagonal element i, i is the sum of the squares of the gradients w.r.t. x_i up to time step t, while ϵ (e.g. 10^{-8}) is a smoothing term that avoids division by zero. In vector form, we have

$$\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - \frac{\eta}{\sqrt{G^t + \epsilon}} \odot \boldsymbol{g}^t.$$

Adagrad eliminates the tuning of learning rate, but $G^t + \epsilon$ can go to infinity.

Other popular SGD variants: Adadelta or RMSprop

Instead of taking the sum of the squares of the gradients w.r.t. x_i up to time step t, Adadelta or RMSprop takes the following moving average

$$E[\boldsymbol{g}^t]^2 = \gamma E[\boldsymbol{g}^{t-1}]^2 + (1-\gamma)[\boldsymbol{g}^t]^2,$$

where $0 < \gamma < 1$ is a constant, and then we update $\boldsymbol{x}^t$ as follows:

$$\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - \frac{\eta}{\sqrt{E[\boldsymbol{g}^t]^2 + \epsilon}}\boldsymbol{g}^t.$$

Other popular SGD variants: Adam

Adam further integrates momentum into RMSprop. In particular, we compute the decaying averages of past and past squared gradients $\boldsymbol{m}^t$ and $\boldsymbol{v}^t$ as follows:

$$\boldsymbol{m}^t = \beta_1 \boldsymbol{m}^{t-1} + (1-\beta_1)\boldsymbol{g}^t; \;\; \boldsymbol{v}^t = \beta_2 \boldsymbol{v}^{t-1} + (1-\beta_2)[\boldsymbol{g}^t]^2,$$

where we typically set $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Next, we do the following bias-correction

$$\hat{\boldsymbol{m}}^t = \frac{\boldsymbol{m}^t}{1-\beta_1^t}; \;\; \hat{\boldsymbol{v}}^t = \frac{\boldsymbol{v}^t}{1-\beta_2^t}.$$

Finally, we update the parameter as follows:

$$\boldsymbol{x}^{t+1} = \boldsymbol{x}^t - \frac{\eta}{\sqrt{\hat{\boldsymbol{v}}^t + \epsilon}} \hat{\boldsymbol{m}}^t.$$

Batch Normalization

To facilitate learning, we typically normalize the initial values of our parameters by initializing them with zero mean and unit variance. As training progresses and we update parameters to different extents, we lose this normalization, which slows down training and amplifies changes as the network becomes deeper.

Batch normalization reestablishes these normalizations for every mini-batch and changes are back-propagated through the operation as well. By making normalization part of the model architecture, we are able to use higher learning rates and pay less attention to the initialization parameters. Batch normalization additionally acts as a regularizer.

Early Stopping

Monitor error on a validation set during training and stop (with some patience) if your validation error does not improve enough.