TTIC 31230, Fundamentals of Deep Learning

David McAllester

An SGD Algorithm from Winter 2017

A Quenching SGD Algorithm

Quenching

Steel is a mixture of iron and carbon. At temperatures below 727° C the carbon "freezes out" of the iron and we get iron grains separated by carbon sheets. But above 727^o the carbon sheets "evaporate" into the iron and we get a homogeneous mixture of iron and carbon that is still a crystaline solid but with a different crystal structure (steel melts around 1510°). If we heat steel above 727° and then drop it in water the carbon does not have time to segregate out of the steel and we get "hardened steel" with a different lattice structure from slowly cooled grainy "soft steel". Hardened steel can be used as a cutting blade in a drill bit to drill into soft steel.

Annealing and Tempering

Annealing is a process of gradually reducing the temperature. Gradual annealing produces soft grainy steel.

Tempering is a process of re-heating quenched steel to temperatures high enough to change its properties but below the original pre-quenching temperature. This can make the steel less brittle while preserving its hardness.

Acknowledgments to my eighth grade shop teacher, 1971.

Is Quenching Desirable?

These slides describe an SGD algorithm that I designed at a time when I assumed that quenching was desirable — that one should design SGD to reach a local minimum as quickly as possible.

A Quenching Algorithm

Suppose that we want to quench the parameters — to reach a local minimum as quickly as possible.

We must consider

- **Gradient Estimation.** The accuracy of $\hat{g}$ as an estimate of g.

- **Gradient Drift (second order structure).** The fact that g changes as the parameters change.

Analysis Plan

We will calculate a batch size B^* and learning rate η^* by optimizing an improvement guarantee for a single batch update.

We then use learning rate scaling to derive the learning rate η_B for a batch size $B << B^*$.

Deriving Learning Rates

If we can calculate B^* and η^* for optimal loss reduction in a single batch we can calculate η_B.

$$\eta_B = B\ \eta_1$$

$$\eta^* = B^* \eta_1$$

$$\eta_1 = \frac{\eta^*}{B^*}$$

$$\eta_B = \frac{B}{B^*}\ \eta^*$$

Calculating B^* and η^* in One Dimension

We will first calculate values B^* and η^* by optimizing the loss reduction over a single batch update in one dimension.

$$g = \hat{g} \pm \frac{2\hat{\sigma}}{\sqrt{B}}$$

$$\hat{\sigma} = \sqrt{E_{(x,y)\sim\mathrm{Batch}}\left(\frac{d\,\mathrm{loss}(\beta, x, y)}{d\beta} - \hat{g}\right)^2}$$

The Second Derivative of $\text{loss}(\beta)$

$$\text{loss}(\beta) \;=\; E_{(x,y)\sim\text{Train}}\;\text{loss}(\beta, x, y)$$

$$d^2\text{loss}(\beta)/d\beta^2 \;\leq\; L \quad \text{(Assumption)}$$

$$\text{loss}(\beta - \Delta\beta) \;\leq\; \text{loss}(\beta) - g\Delta\beta + \frac{1}{2}L\Delta\beta^2$$

$$\text{loss}(\beta - \eta\hat{g}) \;\leq\; \text{loss}(\beta) - g(\eta\hat{g}) + \frac{1}{2}L(\eta\hat{g})^2$$

A Progress Guarantee

$$
\begin{aligned}
\operatorname{loss}(\beta - \eta\hat{g}) &\leq \operatorname{loss}(\beta) - g(\eta\hat{g}) + \frac{1}{2}L(\eta\hat{g})^2 \\
&= \operatorname{loss}(\beta) - \eta(\hat{g} - (\hat{g} - g))\hat{g} + \frac{1}{2}L\eta^2\hat{g}^2 \\
&\leq \operatorname{loss}(\beta) - \eta\left(\hat{g} - \frac{2\hat{\sigma}}{\sqrt{B}}\right)\hat{g} + \frac{1}{2}L\eta^2\hat{g}^2
\end{aligned}
$$

Optimizing B and η

$$\mathrm{loss}(\beta - \eta\hat{g}) \leq \mathrm{loss}(\beta) - \eta\left(\hat{g} - \frac{2\hat{\sigma}}{\sqrt{B}}\right)\hat{g} + \frac{1}{2}L\eta^2\hat{g}^2$$

We optimize progress per gradient calculation by optimizing the right hand side divided by B. The derivation at the end of the slides gives

$$B^* = \frac{16\hat{\sigma}^2}{\hat{g}^2}, \qquad \eta^* = \frac{1}{2L}$$

$$\eta_B = \frac{B}{B^*}\eta^* = \frac{B\hat{g}^2}{32\hat{\sigma}^2 L}$$

Recall this is all just in one dimension.

Estimating $\hat{g}_{B^*}$ and $\hat{\sigma}_{B^*}$

$$\eta_B = \frac{B\hat{g}^2}{32\hat{\sigma}^2 L}$$

We are left with the problem that $\hat{g}$ and $\hat{\sigma}$ are defined in terms of batch size $B^* >> B$.

We can estimate $\hat{g}_{B^*}$ and $\hat{\sigma}_{B^*}$ using a running average with a time constant corresponding to B^*.

Estimating $\hat{g}_{B^*}$

$$\hat{g}_{B^*} = \frac{1}{B^*} \sum_{(x,y)\sim \text{Batch}(B^*)} \frac{d\ \text{Loss}(\beta, x, y)}{d\beta}$$

$$= \frac{1}{N} \sum_{s=t-N+1}^{t} \hat{g}^s \qquad \text{with } N = \frac{B^*}{B} \text{ for batch size } B$$

$$\tilde{g}^{t+1} = \left(1 - \frac{B}{B^*}\right) \tilde{g}^t + \frac{B}{B^*} \hat{g}^{t+1}$$

We are still working in just one dimension.

A Complete Calculation of η (in One Dimension)

$$
\begin{aligned}
\tilde{g}^{t+1} &= \left(1 - \frac{B}{B*(t)}\right)\tilde{g}^t + \frac{B}{B*(t)}\hat{g}^{t+1} \\
\tilde{s}^{t+1} &= \left(1 - \frac{B}{B*(t)}\right)\tilde{s}^t + \frac{B}{B*(t)}(\hat{g}^{t+1})^2 \\
\tilde{\sigma}^t &= \sqrt{\tilde{s}^t - (\tilde{g}^t)^2} \\
B^*(t) &= \begin{cases} K & \text{for } t \leq K \\ 16(\tilde{\sigma}^t)^2/((\tilde{g}^t)^2 + \epsilon) & \text{otherwise} \end{cases}
\end{aligned}
$$

A Complete Calculation of η (in One Dimension)

$$\eta^t = \begin{cases} 0 & \text{for } \; t \leq K \\ \frac{(\tilde{g}^t)^2}{32(\tilde{\sigma}^t)^2 L} & \text{otherwise} \end{cases}$$

As $t \rightarrow \infty$ we expect $\tilde{g}^t \rightarrow 0$ and $\tilde{\sigma}^t \rightarrow \sigma > 0$ which implies $\eta^t \rightarrow 0$.

The High Dimensional Case

So far we have been considering just one dimension.

We now propose treating each dimension $\Phi[i]$ of a high dimensional parameter vector Φ independently using the one dimensional analysis.

We can calculate $B^*[i]$ and $\eta^*[i]$ **for each individual parameter** $\Phi[i]$.

Of course the actual batch size B will be the same for all parameters.

A Complete Algorithm

$$\tilde{g}^{t+1}[i] = \left(1 - \frac{B}{B^*(t)[i]}\right)\tilde{g}^t[i] + \frac{B}{B^*(t)[i]}\hat{g}^{t+1}[i]$$

$$\tilde{s}^{t+1}[i] = \left(1 - \frac{B}{B^*(t)[i]}\right)\tilde{s}^t[i] + \frac{B}{B^*(t)[i]}\hat{g}^{t+1}[i]^2$$

$$\tilde{\sigma}^t[i] = \sqrt{\tilde{s}^t[i] - \tilde{g}^t[i]^2}$$

$$B^*(t)[i] = \begin{cases} K & \text{for } \ t \leq K \\ \lambda_B \tilde{\sigma}^t[i]^2/(\tilde{g}^t[i]^2 + \epsilon) & \text{otherwise} \end{cases}$$

A Complete Algorithm

$$\eta^t[i] = \begin{cases} 0 & \text{for } \; t \leq K \\ \frac{\lambda_\eta \tilde{g}^t[i]^2}{\tilde{\sigma}^t[i]^2} & \text{otherwise} \end{cases}$$

$$\Phi^{t+1}[i] = \Phi^t[i] - \eta^t[i]\hat{g}^t[i]$$

Here we have meta-parameters K, λ_B, ϵ and λ_η.

Appendix: Optimizing B and η

$$\mathrm{loss}(\beta - \eta\hat{g}) \leq \mathrm{loss}(\beta) - \eta\hat{g}\left(\hat{g} - \frac{2\hat{\sigma}}{\sqrt{B}}\right) + \frac{1}{2}L\eta^2\hat{g}^2$$

Optimizing η we get

$$\hat{g}\left(\hat{g} - \frac{2\hat{\sigma}}{\sqrt{B}}\right) = L\eta\hat{g}^2$$

$$\eta^*(B) = \frac{1}{L}\left(1 - \frac{2\hat{\sigma}}{\hat{g}\sqrt{B}}\right)$$

Inserting this into the guarantee gives

$$\mathrm{loss}(\Phi - \eta\hat{g}) \leq \mathrm{loss}(\Phi) - \frac{L}{2}\eta^*(B)^2\hat{g}^2$$

Optimizing B

Optimizing progress per sample, or maximizing $\eta^*(B)^2/B$, we get

$$
\begin{aligned}
\frac{\eta^*(B)^2}{B} &= \frac{1}{L^2}\left(\frac{1}{\sqrt{B}} - \frac{2\hat{\sigma}}{\hat{g}B}\right)^2 \\
0 &= -\frac{1}{2}B^{-\frac{3}{2}} + \frac{2\hat{\sigma}}{\hat{g}}B^{-2} \\
B^* &= \frac{16\hat{\sigma}^2}{\hat{g}^2} \\
\eta^*(B^*) = \eta^* &= \frac{1}{2L}
\end{aligned}
$$

Appendix II: A Formal Bound for the Vector Case

We will prove that minibatch SGD for a **sufficiently large batch size** (for gradient estimation) and a **sufficient small learning rate** (to avoid gradient drift) is guaranteed (with high probability) to reduce the loss.

This guarantee has two main requirements.

- A smoothness condition to limit gradient drift.
- A bound on the gradient norm allowing high confidence gradient estimation.

Smoothness: The Hessian

We can make a second order approximation to the loss.

$$\ell(\Phi + \Delta\Phi) \approx \ell(\Phi) + g^\top \Delta\Phi + \frac{1}{2}\Delta\Phi^\top H \Delta\Phi$$

$$g = \nabla_\Phi \, \ell(\Phi)$$

$$H = \nabla_\Phi \nabla_\Phi \, \ell(\Phi)$$

The Smoothness Condition

We will assume

$$||H\Delta\Phi|| \leq L||\Delta\Phi||$$

We now have

$$\Delta\Phi^\top H\Delta\Phi \leq L||\Delta\Phi||^2$$

Using the second order mean value theorem one can prove

$$\ell(\Phi + \Delta\Phi) \;\leq\; \ell(\Phi) + g^\top \Delta\Phi + \frac{1}{2}L||\Delta\Phi||^2$$

A Concentration Inequality for Gradient Estimation

Consider a vector mean estimator where the vectors g_n are drawn IID.

$$g_n = \nabla_\Phi \ell_n(\Phi) \qquad \hat{g} = \frac{1}{k}\sum_{n=1}^{k} g_n \qquad g = E_n\, \nabla_\Phi\, \ell_n(\Phi)$$

If with probability 1 over the draw of n we have $|(g_n)_i - g_i| \leq b$ for all i then with probability of at least $1-\delta$ over the draw of the sample

$$||\hat{g} - g|| \leq \frac{\eta}{\sqrt{k}} \qquad \eta = b\left(1 + \sqrt{2\ln(1/\delta)}\right)$$

Norkin and Wets "Law of Small Numbers as Concentration Inequalities ...", 2012, theorem 3.1

$$
\begin{aligned}
\ell(\Phi + \Delta\Phi) &\leq \ell(\Phi) + g^\top \Delta\Phi + \frac{1}{2} L ||\Delta\Phi||^2 \\
\ell(\Phi - \eta\widehat{g}) &\leq \ell(\Phi) - \eta g^\top \widehat{g} + \frac{1}{2} L\eta^2 ||\widehat{g}||^2 \\
&= \ell(\Phi) - \eta(\widehat{g} - (\widehat{g} - g))^\top \widehat{g} + \frac{1}{2} L\eta^2 ||\widehat{g}||^2 \\
&= \ell(\Phi) - \eta||\widehat{g}||^2 + \eta(\widehat{g} - g)^\top \widehat{g} + \frac{1}{2} L\eta^2 ||\widehat{g}||^2 \\
&\leq \ell(\Phi) - \eta||\widehat{g}||^2 + \eta\frac{\eta}{\sqrt{k}} ||\widehat{g}|| + \frac{1}{2} L\eta^2 ||\widehat{g}||^2 \\
&= \ell(\Phi) - \eta||\widehat{g}|| \left(||\widehat{g}|| - \frac{\eta}{\sqrt{k}} \right) + \frac{1}{2} L\eta^2 ||\widehat{g}||^2
\end{aligned}
$$

Optimizing η

Optimizing η we get

$$||\widehat{g}|| \left(||\widehat{g}|| - \frac{\eta}{\sqrt{k}} \right) = -L\eta ||\widehat{g}||^2$$

$$\eta = \frac{1}{L} \left(1 - \frac{\eta}{||\widehat{g}||\sqrt{k}} \right)$$

Inserting this into the guarantee gives

$$\ell(\Phi - \eta\widehat{g}) \leq \ell(\Phi) - \frac{L}{2}\eta^2 ||\widehat{g}||^2$$

Optimizing k

Optimizing progress per sample, or maximizing η^2/k, we get.

$$\frac{\eta^2}{k} = \frac{1}{L^2}\left(\frac{1}{\sqrt{k}} - \frac{2\hat{\sigma}}{||\widehat{g}||k}\right)^2$$

$$0 = -\frac{1}{2}k^{-\frac{3}{2}} + \frac{2\hat{\sigma}}{||\widehat{g}||}k^{-2}$$

$$k = \left(\frac{22\hat{\sigma}}{||\widehat{g}||}\right)^2$$

$$\eta = \frac{1}{2L}$$

END