Stochastic Gradient Descent

Compute gradient estimate

$$\hat{\mathbf{g}} \leftarrow +\frac{1}{m}\nabla_{\boldsymbol{\theta}} \sum_i L(f(\mathbf{x}_i;\boldsymbol{\theta}),\mathbf{y}_i)$$

Apply update

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \epsilon\hat{\mathbf{g}}$$

Use sample randomly or in random batches instead of using complete data at each update

Depend only on local gradient

Note: Reduce Dependency

Challenge: Non Convex Problem, Slow Convergence

Momentum

Compute gradient estimate

$$\hat{\mathbf{g}} \leftarrow +\frac{1}{m}\nabla_{\boldsymbol{\theta}} \sum_i L(f(\mathbf{x}_i;\boldsymbol{\theta}),\mathbf{y}_i)$$

Compute velocity update

$$\mathbf{v} \leftarrow \alpha\mathbf{v} - \epsilon\hat{\mathbf{g}}$$

Apply update

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \mathbf{v}$$

Faster near minima, avoid slow convergence

Note: Faster Convergence, Reduced Oscillation

Challenge: Blindly follow slops

Nesterov momentum

Compute interim update

$$\widehat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta} + \alpha\mathbf{v}$$

Compute gradient (at interim point)

$$\hat{\mathbf{g}} \leftarrow +\frac{1}{m}\nabla_{\widehat{\boldsymbol{\theta}}} \sum_i L\left(f\left(\mathbf{x}_i;\widehat{\boldsymbol{\theta}}\right),\mathbf{y}_i\right)$$

Compute velocity update

$$\mathbf{v} \leftarrow \alpha\mathbf{v} - \epsilon\hat{\mathbf{g}}$$

Apply update

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \mathbf{v}$$

Note: Faster Convergence, Know where it is going

Challenge: Not Adaptive

AdaGrad

Compute gradient estimate

$$\hat{\mathbf{g}} \leftarrow +\frac{1}{m}\nabla_{\boldsymbol{\theta}} \sum_i L(f(\mathbf{x}_i;\boldsymbol{\theta}),\mathbf{y}_i)$$

Accumulate squared gradient

$$\mathbf{r} \leftarrow \mathbf{r} + \hat{\mathbf{g}} \odot \hat{\mathbf{g}}$$

Compute parameter update (Division and square root applied element-wise)

$$\Delta\boldsymbol{\theta} \leftarrow -\frac{\epsilon}{\delta + \sqrt{\mathbf{r}}} \odot \hat{\mathbf{g}}$$

Apply update

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$$

Learning rate is adaptive, slows down near minima

Note: Adaptive

Challenge: Keeps going, Learning Rate shrinks

RMSProp

Compute gradient estimate

$$\hat{\mathbf{g}} \leftarrow +\frac{1}{m}\nabla_{\boldsymbol{\theta}} \sum_i L(f(\mathbf{x}_i;\boldsymbol{\theta}),\mathbf{y}_i)$$

Note: Recursive, LR optimally high

Challenge: average of past gradients

Accumulate squared gradient

$$\mathbf{r} \leftarrow \rho \mathbf{r} + (1-\rho)\hat{\mathbf{g}} \odot \hat{\mathbf{g}}$$

Compute parameter update (Division and square root applied element-wise)

$$\Delta\boldsymbol{\theta} \leftarrow -\frac{\epsilon}{\delta + \sqrt{\mathbf{r}}} \odot \hat{\mathbf{g}}$$

Apply update

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$$

RMSProp with Nesterov momentum

Compute interim update

$$\widehat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta} + \alpha \mathbf{v}$$

Compute gradient (at interim point)

$$\hat{\mathbf{g}} \leftarrow +\frac{1}{m} \nabla_{\widehat{\boldsymbol{\theta}}} \sum_i L\left(f\left(\mathbf{x}_i; \widehat{\boldsymbol{\theta}}\right), \mathbf{y}_i\right)$$

Accumulate squared gradient

$$\mathbf{r} \leftarrow \rho \mathbf{r} + (1-\rho)\hat{\mathbf{g}} \odot \hat{\mathbf{g}}$$

Compute velocity update

$$\mathbf{v} \leftarrow \alpha \mathbf{v} - \frac{\epsilon}{\delta + \sqrt{\mathbf{r}}} \odot \hat{\mathbf{g}}$$

Apply update

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \mathbf{v}$$

Use two knobs to adapt learning

Adam

Note: adaptive

Compute gradient estimate

$$\hat{\mathbf{g}} \leftarrow +\frac{1}{m} \nabla_{\boldsymbol{\theta}} \sum_i L(f(\mathbf{x}_i; \boldsymbol{\theta}), \mathbf{y}_i)$$

$$t \leftarrow t + 1$$

Update biased first moment estimate

$$\mathbf{s}_t \leftarrow \rho_1 \mathbf{s}_t + (1-\rho_1)\hat{\mathbf{g}}$$

Update biased second moment estimate

$$\mathbf{r}_t \leftarrow \rho_2 \mathbf{r}_t + (1-\rho_2)\hat{\mathbf{g}} \odot \hat{\mathbf{g}}$$

Correct bias in first moment

$$\widehat{\mathbf{s}_t} \leftarrow \frac{\mathbf{s}_t}{1-\rho_1^t}$$

Correct bias in second moment

$$\widehat{\mathbf{r}_t} \leftarrow \frac{\mathbf{r}_t}{1-\rho_2^t}$$

Compute parameter update (Division and square root applied element-wise)

$$\Delta\boldsymbol{\theta} \leftarrow -\frac{\epsilon}{\delta + \sqrt{\widehat{\mathbf{r}_t}}} \odot \widehat{\mathbf{s}_t}$$

Apply update

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$$

Use same rule for each step, no special case for initialization

Ref. Book: Chapter 8.3 & 8.5, Deep Learning. Ian Goodfellow, Yoshua Bengio and Aaron Courville