Advanced Machine Learning

Likhit Nayak

Gradient descent

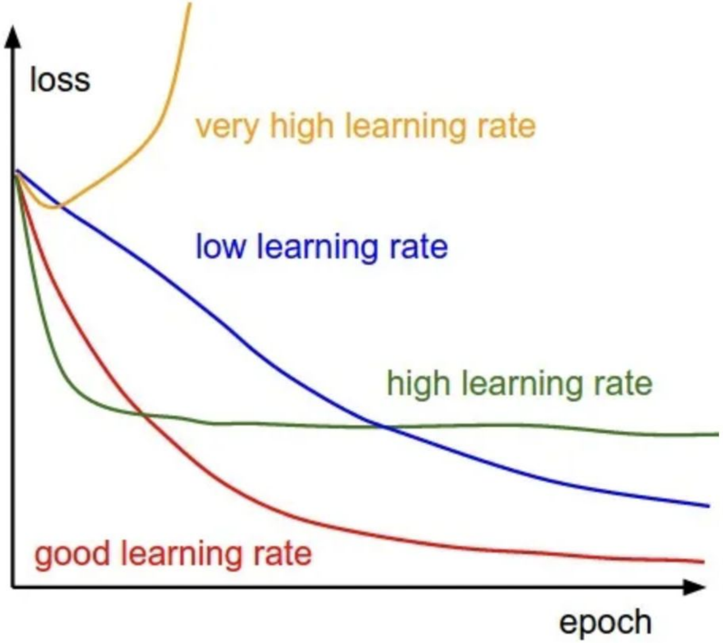
$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta)$$

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x^{(i)}; y^{(i)})$$

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x^{(i:i+n)}; y^{(i:i+n)})$$

Disadvantages of SGD

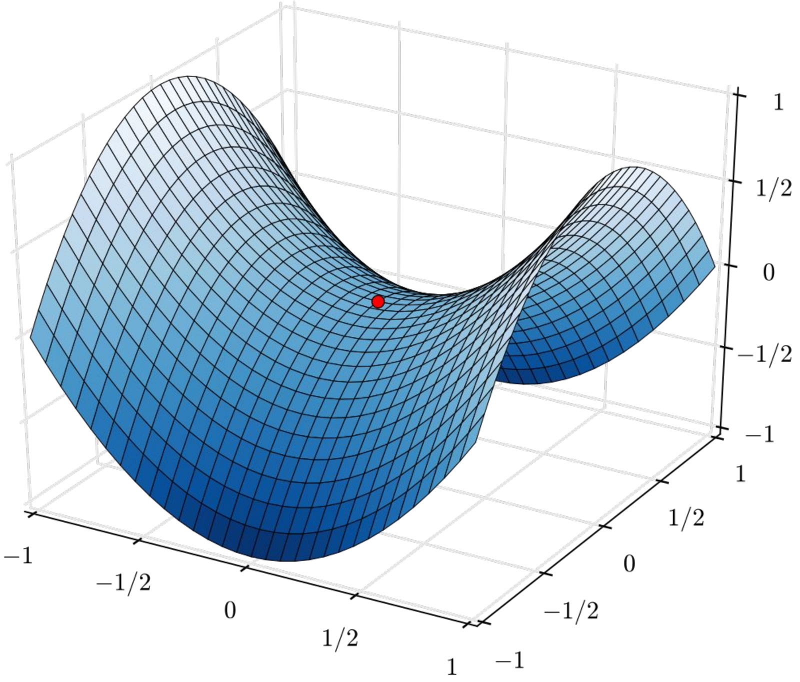
1. Dependent on learning rate



https://cs231n.github.io/neural-networks-3/

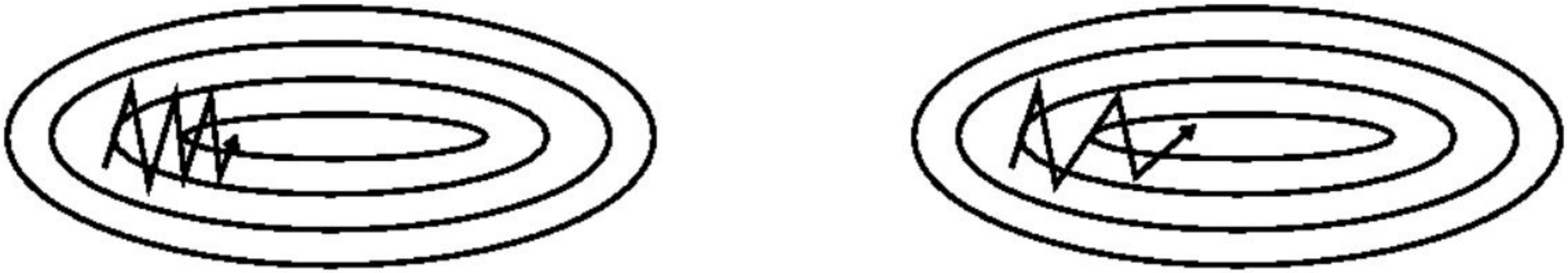
Disadvantages of SGD

2. Learning rate schedulers and thresholds defined in advance are unable to adapt to a dataset's characteristics
3. The same learning rate applies to all parameter updates
    - If our data/feature set is sparse, we might want to different updates for different features
4. Doesn't know how to deal with saddle points, i.e. points where one dimension slopes up and another slopes down



Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).

SGD with momentum



(a) SGD without momentum

(b) SGD with momentum

Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).

SGD with momentum

Momentum helps accelerate SGD in the relevant direction and dampens oscillations by adding another hyperparameter ***γ***:

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta J(\theta)$$

$$\theta = \theta - v_t$$

Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).

SGD with Nesterov momentum

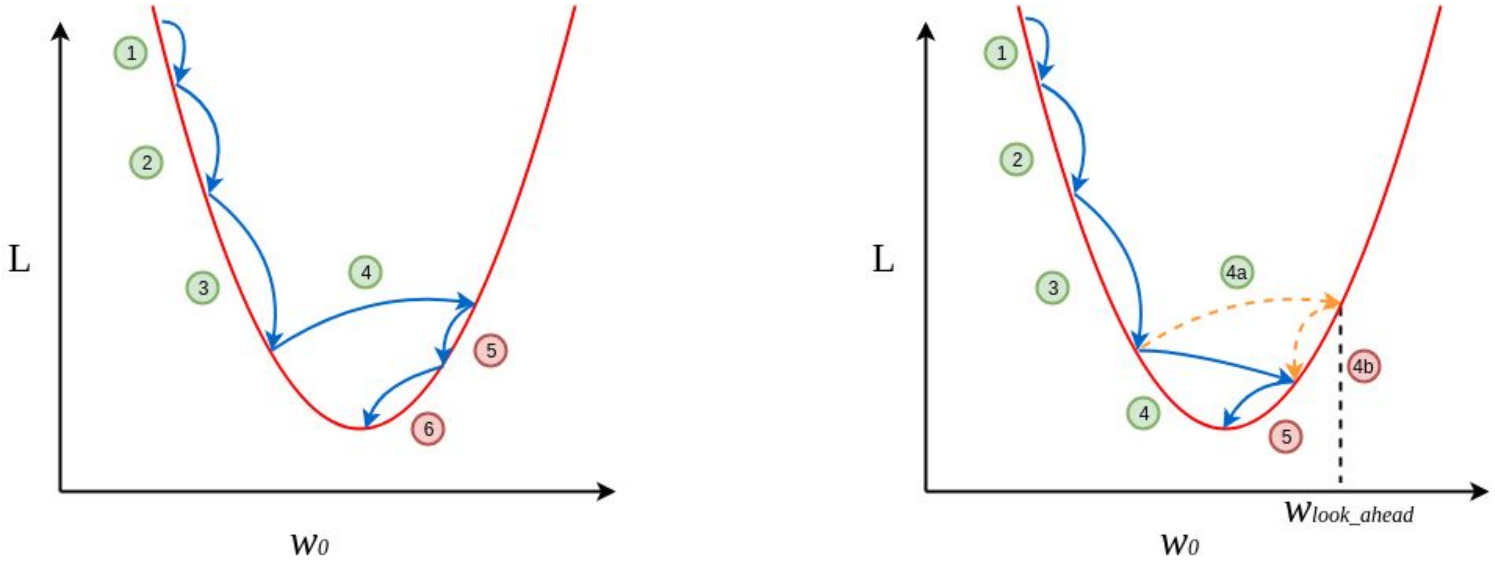
Computes an additional term representing the approximate future position of the parameters ***θ***:

$$v_t = \gamma\, v_{t-1} + \eta \nabla_\theta J(\theta - \gamma v_{t-1})$$

$$\theta = \theta - v_t$$

Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).

SGD with Nesterov momentum



(a) Momentum-Based Gradient Descent

(b) Nesterov Accelerated Gradient Descent

Adagrad

It adapts the learning rate to the parameters, performing larger updates for infrequent and smaller updates for frequent parameters

If we consider SGD for each single parameter $\boldsymbol{\theta}_i$:

$$g_{t,i} = \nabla_{\theta_t} J(\theta_{t,i})$$

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} \cdot g_{t,i}$$

Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).

RMSProp

Instead of accumulating the sum of squared gradients, this method uses a decaying average of past gradients:

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma) g_t^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).

Adam

In addition to using the past squared gradients, Adaptive Moment Estimation (Adam) also uses past gradients:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

Adam

In addition to using the past squared gradients, Adaptive Moment Estimation (Adam) also uses past gradients:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

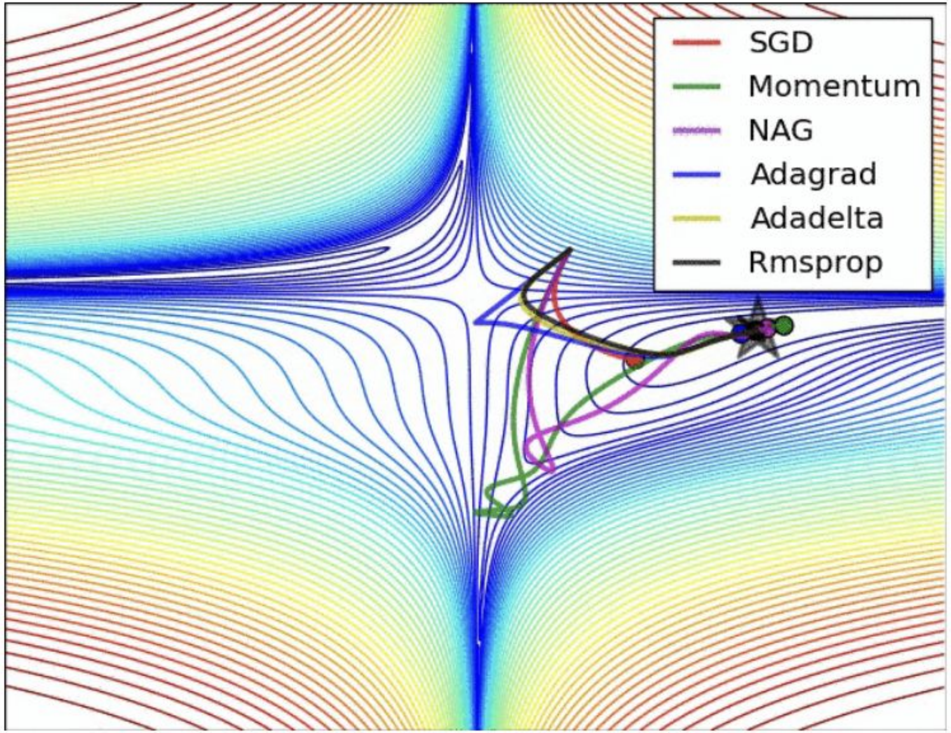
$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

Adam

In addition to using the past squared gradients, Adaptive Moment Estimation (Adam) also uses past gradients:

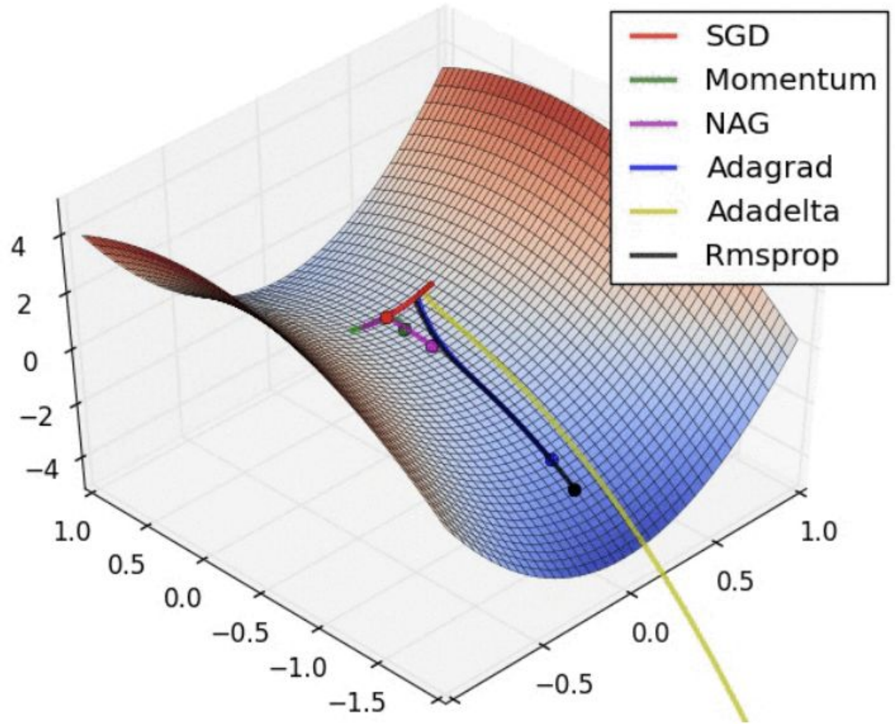
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

Comparison of SGD variants



Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).

Comparison of SGD variants



Ruder, Sebastian. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* (2016).