Idea Factory Intensive Program #2

딥러닝 홀로서기

#17

이론강의/PyTorch실습/코드리뷰

딥러닝(Deep Learning)에 관심이 있는 학생 발굴을 통한
딥러닝의 이론적 배경 강의 및 오픈소스 딥러닝 라이브러리 PyTorch를 활용한 실습

Topics to learn today

1. Review from last lecture

Assignment: CIFAR-10 classification with Pytorch

Lecture: Overcoming overfitting and gradient vanishing

2. Batch/Stochastic Gradient Descent

3. Advanced Gradient Descent Algorithms

Momentum, NAG, AdaGrad, AdaDelta, RMSProp, ADAM

4. How to visualize the result

Pandas DataFrame, Seaborn

Review from Last Lecture

Overfitting: L2 Regularization / Dropout

Review from Last Lecture

Gradient Vanishing: ReLU Activation

Review from Last Lecture

Xavier Initialization / Batch Normalization

Batch/Stochastic Gradient Descent

Gradient Descent

$$\theta = \theta - \eta \nabla J(\theta)$$

θ : Parameter set of the model η : Learning rate $J(\theta)$: Loss function

Batch Gradient Descent

Calculate gradient of parameters for whole training dataset.

Need a lot of memory depending on data.
Calculating gradient is too slow, thus optimization is slow.

Stochastic Gradient Descent (SGD)

Calculate gradient for small chunk of whole training dataset (mini-batch), rather than the whole training dataset (batch).

Stochastic since the gradient is not deterministic, but stochastic depending on the mini-batch.

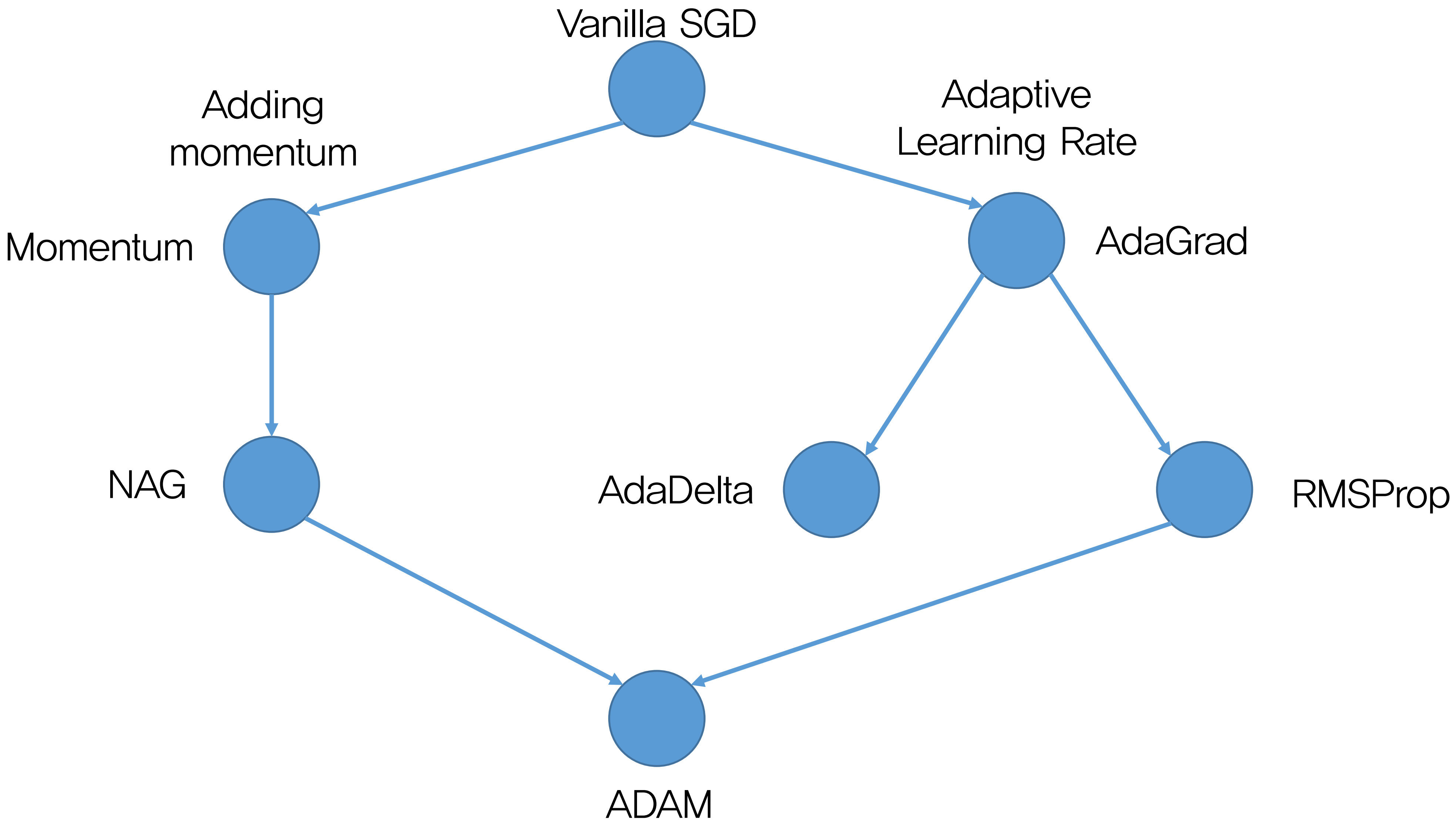
Faster than batch gradient descent, while converging similar.

Can avoid local minima by stochasticity.

Advanced Gradient Descent Methods

http://shuuki4.github.io/deep%20learning/2016/05/20/Gradient-Descent-Algorithm-Overview.html

Diagram of Gradient Descent Development



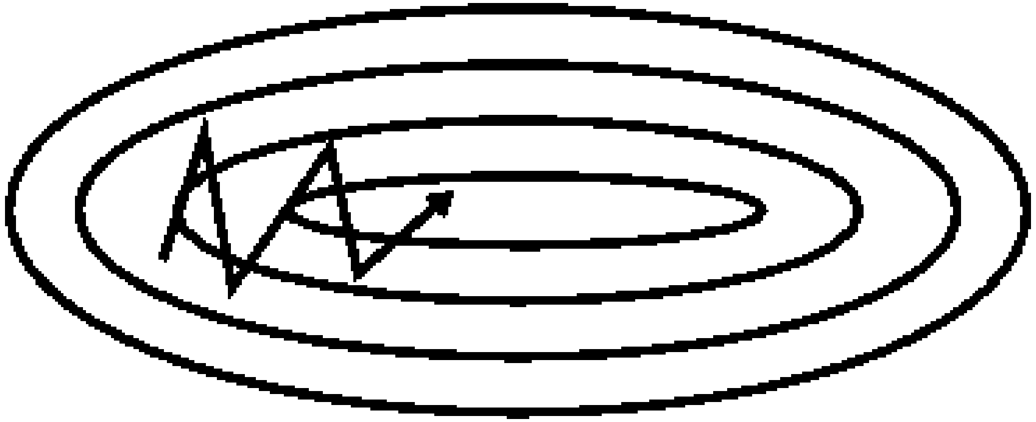
Problem of Vanilla SGD

Cannot escape local minima.

Momentum

$$\theta = \theta - v_t$$

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



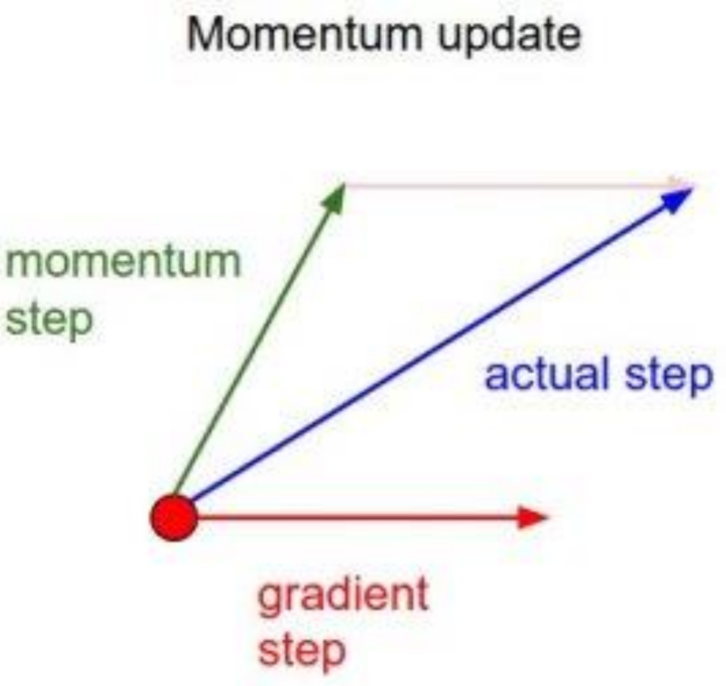
Problem of Momentum

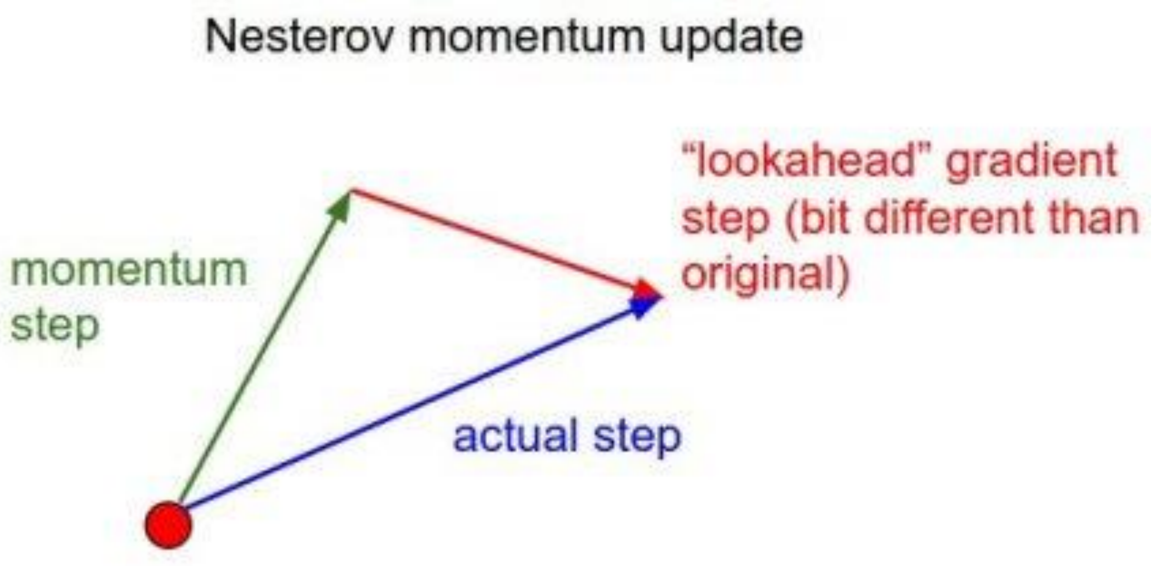
Can escape local minima, but cannot stop or slow at global minima.

Nesterov Accelerated Gradient (NAG)

$$\theta = \theta - v_t$$

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





Stanford, cs231n

Problem of NAG

Step size is equal for every parameter

Adaptive Gradient (Adagrad)

$$\theta_{t+1} = \theta - \frac{\eta}{\sqrt{G_t + \epsilon}} \cdot \nabla_\theta J(\theta_t)$$

$$G_t = G_{t-1} + (\nabla_\theta J(\theta_t))^2$$

Problem of Adagrad

G keep increases, thus step size decays to zero

RMSProp

$$\theta_{t+1} = \theta - \frac{\eta}{\sqrt{G_t + \epsilon}} \cdot \nabla_\theta J(\theta_t)$$

$$G_t = \gamma G_{t-1} + (1-\gamma)(\nabla_\theta J(\theta_t))^2$$

AdaDelta

$$\theta_{t+1} = \theta_t - \Delta_\theta$$

$$\Delta_\theta = \frac{\sqrt{s + \epsilon}}{\sqrt{G + \epsilon}} \cdot \nabla_\theta J(\theta_t)$$

$$s_{t+1} = \gamma s_t + (1 - \gamma)\Delta_\theta$$

$$G_{t+1} = \gamma G_t + (1 - \gamma)(\nabla_\theta J(\theta_t))^2$$

AdaDelta

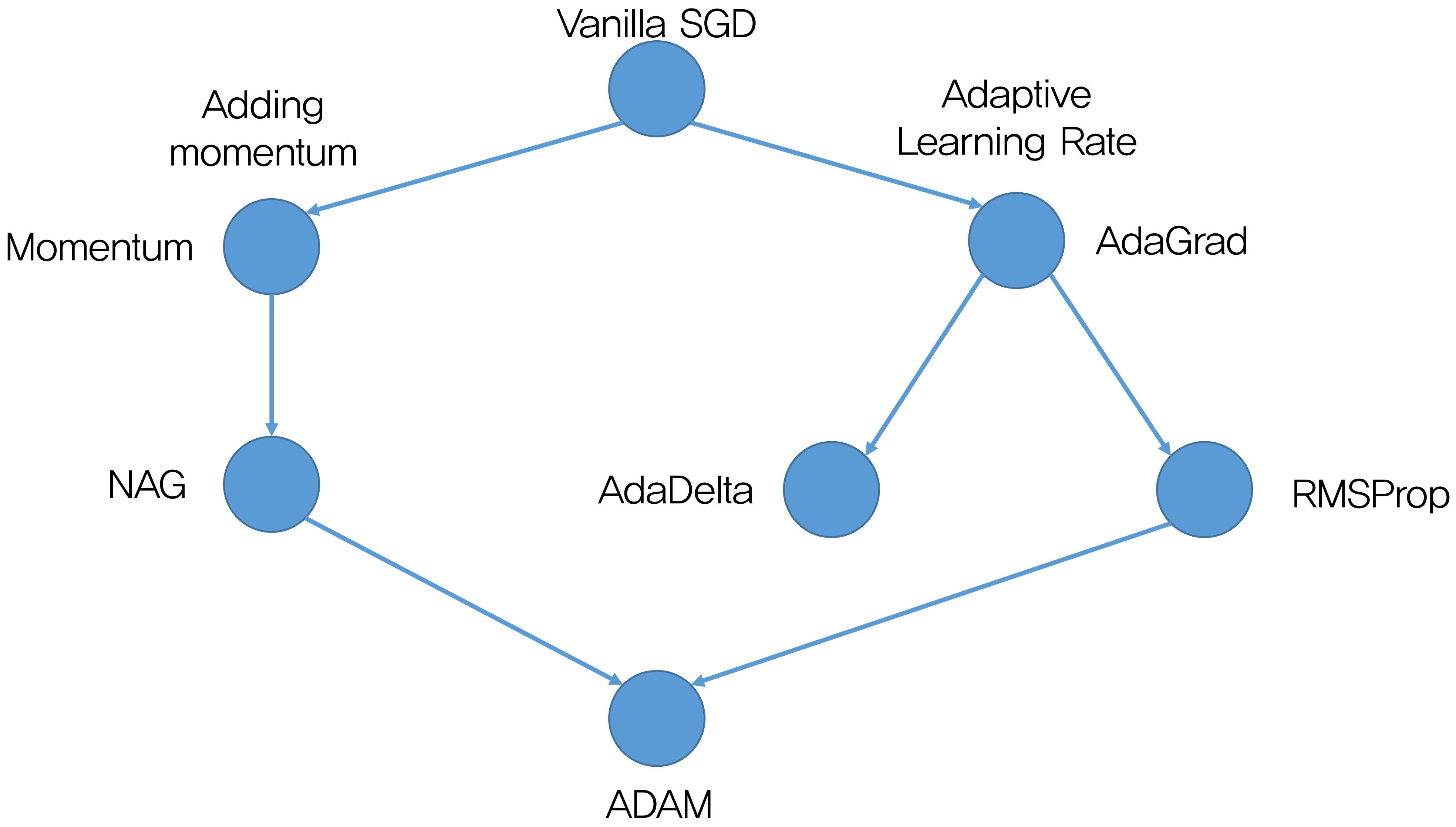
Adaptive Moment Estimation (ADAM)

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

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \qquad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

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

Diagram of Gradient Descent Development



How to use Advanced Optimizers in Pytorch

How to use Advanced Optimizers in Pytorch

```
optimizer = optim.SGD(model.parameters(), lr = 0.01, momentum=0.9)
```