Boulder

Training Deep Neural Networks

YouTube Playlist

Maziar Raissi

Assistant Professor

Department of Applied Mathematics

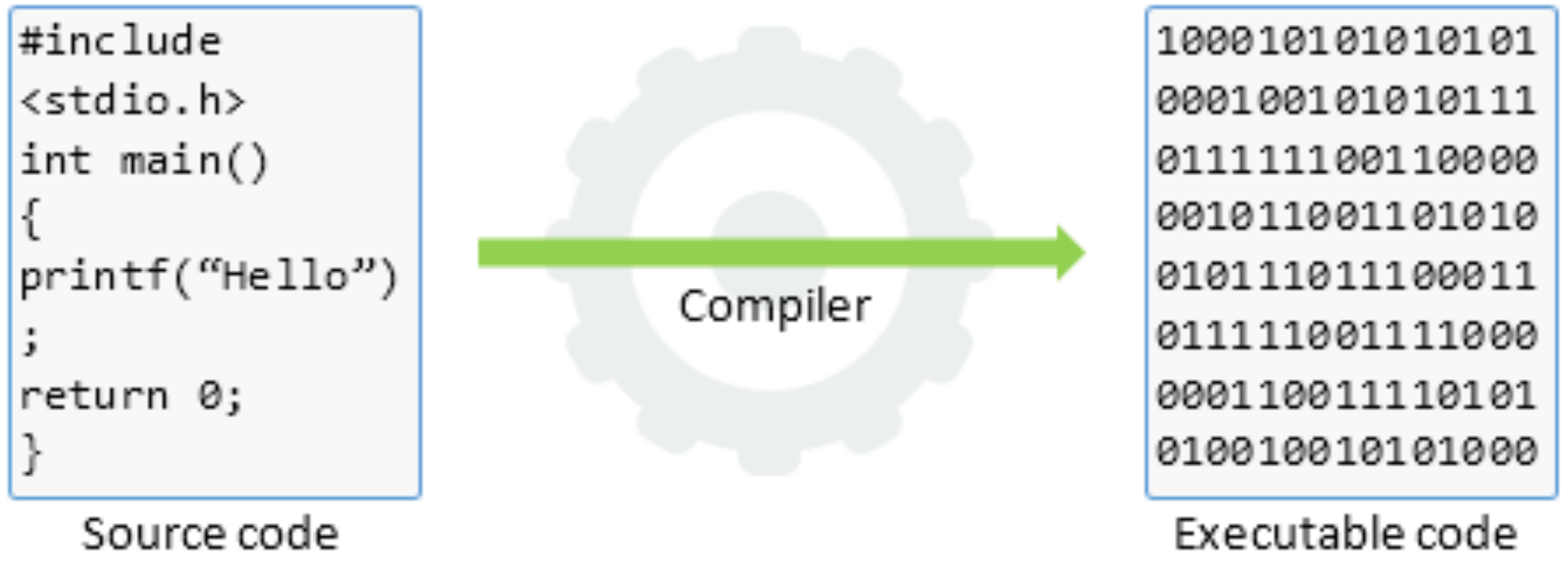
University of Colorado Boulder

maziar.raissi@colorado.edu

Boulder

An overview of gradient descent optimization algorithms

YouTube Playlist



Deep Learning: An algorithm that writes an algorithm

Source Code: Data (examples/experiences)
Compiler: Deep Learning
Executable Code: Deployable Model

Deep: Function Compositions $f_L \circ f_{L-1} \circ \dots f_2 \circ f_1$
Learning: Loss Function, Back-propagation, and Gradient Descent

$$\min_\theta L(\theta)$$

$L(\theta) \approx J(\theta) \rightarrow$ noisy estimate of the objective function (e.g., due to mini-batching)

Stochastic Gradient Descent

$$\theta_{t+1} = \theta_t - \gamma \nabla_\theta J(\theta_t)$$

$J : \mathbb{R}^d \rightarrow \mathbb{R}$, $\nabla_\theta J : \mathbb{R}^d \rightarrow \mathbb{R}^d$ — one back-propagation (fast)

$\nabla_\theta J : \mathbb{R}^d \rightarrow \mathbb{R}^d$, $\nabla^2_\theta J : \mathbb{R}^d \rightarrow \mathbb{R}^{d \times d}$ — d back-propagations (slow)

Momentum

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

Nesterov Accelerated Gradient (NAG)

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

Adagrad

$$g_t = \nabla_\theta J(\theta_t)$$
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \odot g_t$$
$$G_t = \sum_{\tau=1}^{t} g_\tau \odot g_\tau$$

Adadelta

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

units don't match

$$\Delta\theta_t = -\frac{RMS[\Delta\theta]_{t-1}}{RMS[g]_t} g_t$$

RMSprop

$$E[g^2]_t = 0.9 E[g^2]_{t-1} + 0.1 g_t^2$$
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

Adam

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$
$$\left.\begin{aligned} \hat{m}_t &= \frac{m_t}{1-\beta_1^t} \\ \hat{v}_t &= \frac{v_t}{1-\beta_2^t} \end{aligned}\right\} \text{bias-corrected}$$
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

AdaMax

$$u_t = \max(\beta_2 u_{t-1}, |g_t|) \qquad \theta_{t+1} = \theta_t - \frac{\eta}{u_t} \hat{m}_t$$

Nadam

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \left(\beta_1 \boxed{\hat{m}_t} + \frac{(1-\beta_1) g_t}{1-\beta_1^t}\right)$$
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \left(\beta_1 \hat{m}_{t-1} + \frac{(1-\beta_1) g_t}{1-\beta_1^t}\right) \rightarrow \text{Adam}$$

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

Questions?