



deeplearning.ai

Setting up your
optimization problem

Gradient Checking

Gradient check for a neural network

Take $W^{[1]}, b^{[1]}, \dots, W^{[L]}, b^{[L]}$ and reshape into a big vector θ .

concatenate

$$J(w^{[1]}, b^{[1]}, \dots, w^{[L]}, b^{[L]}) = J(\theta)$$

Take $dW^{[1]}, db^{[1]}, \dots, dW^{[L]}, db^{[L]}$ and reshape into a big vector $d\theta$.

Is $d\theta$ the gradient of $J(\theta)$?

Gradient checking (Grad check)

$$J(\theta) = J(\theta_1, \theta_2, \theta_3, \dots)$$

for each i :

$$\rightarrow \underline{d\theta_{\text{approx}}[i]} = \frac{J(\theta_1, \theta_2, \dots, \overset{\downarrow}{\theta_i + \epsilon}, \dots) - J(\theta_1, \theta_2, \dots, \overset{\downarrow}{\theta_i - \epsilon}, \dots)}{2\epsilon}$$

$$\approx \underline{d\theta[i]} = \frac{\partial J}{\partial \theta_i} \quad | \quad d\theta_{\text{approx}} \approx d\theta$$

Check

$$\rightarrow \frac{\|d\theta_{\text{approx}} - d\theta\|_2}{\|d\theta_{\text{approx}}\|_2 + \|d\theta\|_2}$$
$$\epsilon = 10^{-7}$$

$$\approx \frac{10^{-7}}{10^{-5}} - \text{great!} \leftarrow$$
$$\rightarrow 10^{-3} - \text{worry.} \leftarrow$$



deeplearning.ai

Setting up your
optimization problem

Gradient Checking
implementation notes

Gradient checking implementation notes

- Don't use in training – only to debug

$$\frac{d\theta_{\text{approx}}[i]}{\uparrow \uparrow} \longleftrightarrow \frac{d\theta[i]}{\uparrow}$$

- If algorithm fails grad check, look at components to try to identify bug.

$$\underline{\frac{db^{[L]}}{d\theta}} \quad \underline{\frac{dW^{[L]}}{d\theta}}$$

- Remember regularization.

$$\underline{J(\theta)} = \frac{1}{n} \sum_i \ell(y^{(i)}, \hat{y}^{(i)}) + \underbrace{\frac{\lambda}{2m} \sum_l \|W^{[l]}\|_F^2}_{\text{grad of } J \text{ w.r.t. } \theta}$$

- Doesn't work with dropout.

$$\underline{J} \quad \underline{\text{keep-prob} = 1.0}$$

- Run at random initialization; perhaps again after some training.

$$\underline{W, b \approx 0}$$