COSC 594/650
Backpropagation

Dongsheng Ding
dongshed@utk.edu

Department of Electrical Engineering and Computer Science
University of Tennessee, Knoxville

Recap: training uses gradients

Loss: $f(\theta)$, θ collects the weights

Gradient: $g = \nabla f(\theta)$

- A GRADIENT STEP

$$\theta \leftarrow \theta - \alpha g, \qquad \alpha > 0 \text{ is the step size}$$

- LECTURE 24
  - ⋆ smooth PL objectives: step $\alpha = 1/L$ shrinks the loss gap linearly

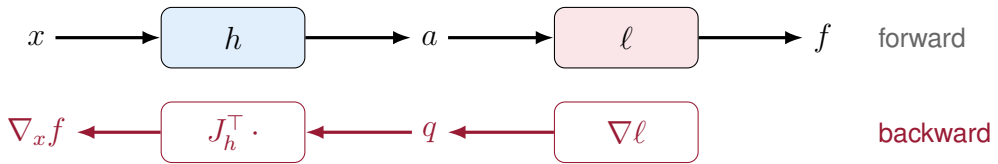
Today: what controls the size of these derivatives?

QUESTION: **Why can deep networks be hard to train even when gradients are easy to compute?**

Chain rule for a composition

Composition: $f(x) = \ell(h(x)), \quad x \in \mathbb{R}^n, \quad a = h(x) \in \mathbb{R}^p, \quad f(x) \in \mathbb{R}$

Local derivatives: Jacobian $J_h = \partial h / \partial x \in \mathbb{R}^{p \times n}$, gradient $q = \nabla_a \ell(a) \in \mathbb{R}^p$



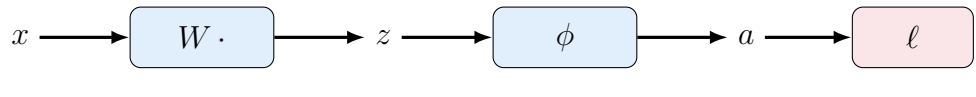
- DIFFERENTIATE h, THEN SUBSTITUTE INTO $df = q^\top da$

$$da = J_h \, dx \quad \Longrightarrow \quad df = q^\top J_h \, dx = \left(J_h^\top q \right)^\top dx$$

$$\nabla_x f = J_h^\top q \quad \textbf{is the chain rule}$$

Backpropagation through one layer

Layer: $z = Wx$, $a = \phi(z)$ coordinatewise, loss $f = \ell(a)$, $q = \nabla_a \ell$



to the layer below

- DIFFERENTIATE $a_j = \phi(z_j)$ AND $z = Wx$

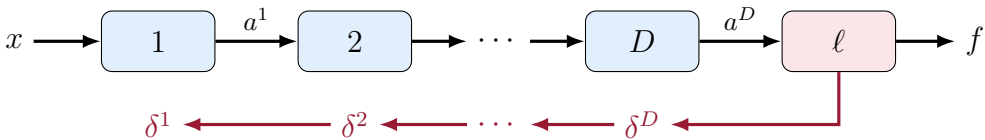
$$\delta = \nabla_z f = \phi'(z) \odot q, \qquad \nabla_x f = W^\top \delta, \qquad \nabla_W f = \delta\, x^\top$$

  - ⋆ $\nabla_W f$ updates W; $\nabla_x f$ is q of the layer below

Backward through a layer: multiply by ϕ', then by $W^\top$

The backpropagation recursion

Network: $a^0 = x, \quad z^l = W^l a^{l-1}, \quad a^l = \phi(z^l)$ for $l < D, \quad a^D = z^D, \quad f = \ell(a^D, y)$



- FROM $\delta^D = \nabla_{a^D} f$ DOWN, $\delta^l = \nabla_{z^l} f$

$$\delta^{l-1} = \phi'(z^{l-1}) \odot \left(W^l\right)^\top \delta^l, \qquad \nabla_{W^l} f = \delta^l \left(a^{l-1}\right)^\top$$

- ⋆ cost: one product per layer forward, two backward: about 3 passes

One backward pass: all D weight gradients

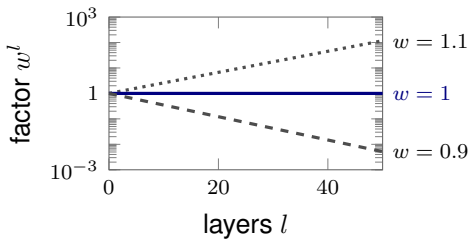
Products of Jacobians control gradient scale

Layer Jacobian: $J_l = \partial a^l / \partial a^{l-1} = \operatorname{diag}\left(\phi'(z^l)\right) W^l, \quad l < D; \quad J_D = W^D$

- REPEAT THE CHAIN RULE

$$\nabla_{a^0} f = J_1^\top J_2^\top \cdots J_D^\top \nabla_{a^D} f, \qquad \|\nabla_{a^0} f\| \le \|J_1\|\|J_2\| \cdots \|J_D\| \, \|\nabla_{a^D} f\|$$

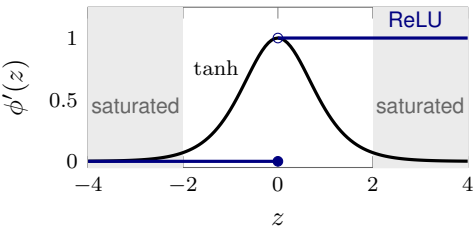
- A SCALAR CHAIN: $J_l = w$
  - ⋆ after l layers: factor w^l
  - ⋆ $0.9^{50} = 0.005, \quad 1.1^{50} = 117$



Depth multiplies factors: gradients can vanish or explode

Activation functions affect the gradient

Backward through layer l: $\nabla_{a^{l-1}} f = J_l^\top \nabla_{a^l} f, \quad J_l = \operatorname{diag}\left(\phi'(z^l)\right) W^l$



- LOCAL DERIVATIVES
  - $\tanh$: $1 - \tanh^2 z \approx 0$ for large $|z|$
  - ReLU: 1 for $z > 0$, 0 for $z < 0$
- BOTH: $0 \le \phi' \le 1$
  - each coordinate can only shrink
- PER LAYER: AT MOST A FACTOR $\|J_l\|$

$$\|\nabla_{a^{l-1}} f\| \;\le\; \|J_l\|\,\|\nabla_{a^l} f\| \;\le\; \max_j \phi'(z_j^l) \cdot \|W^l\| \cdot \|\nabla_{a^l} f\|$$

Activations only shrink; the weights set the scale

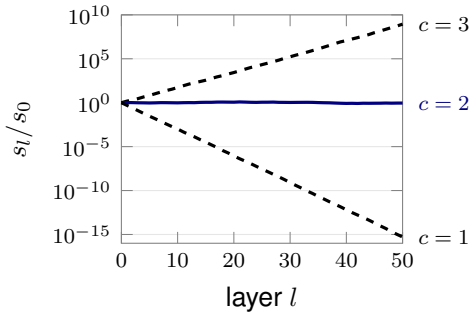
Initialization: an all-ReLU chain

Setup: width n, ReLU, independent weights symmetric about 0, variance c/n

- ONE LAYER: $s_l = \frac{1}{n}\mathbb{E}\|a^l\|^2$

$$s_l = \frac{c}{2}\, s_{l-1}$$

  - ⋆ weighted sum: factor c
  - ⋆ ReLU keeps half: factor $\frac{1}{2}$
- AFTER l LAYERS: $s_l = (c/2)^l\, s_0$
  - ⋆ backward: $\approx$ the same factor

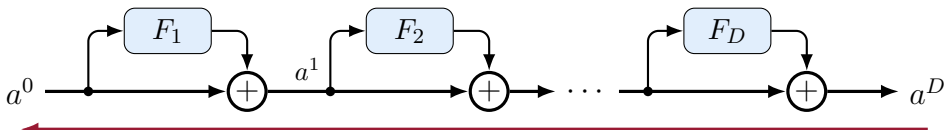


each curve: average of 5 random networks, width 512

Variance $2/n$: each ReLU layer keeps the scale

Residual connections: a direct path for the gradient

Residual block l: $a^l = a^{l-1} + F_l(a^{l-1})$, F_l learned, Jacobian J_{F_l}



BACKWARD THROUGH BLOCK l

$$\nabla_{a^{l-1}} f = \nabla_{a^l} f + J_{F_l}^\top \nabla_{a^l} f$$

- ⋆ direct path: $\nabla_{a^l} f$ comes back with no weight multiplying it
- ⋆ side loop: adds $J_{F_l}^\top \nabla_{a^l} f$, as in a plain layer, can shrink

Direct path: the gradient from the top reaches every layer

Computing and scaling gradients

Question	Answer
How?	from the loss down: multiply by ϕ', then by $(W^l)^\top$ Rumelhart, Hinton, Williams 1986
Cost?	about 3 forward passes, whatever the number of weights Baur, Strassen 1983
Danger?	a product of D layer Jacobians: it can vanish or explode Pascanu, Mikolov, Bengio 2013
Network fixes?	ReLU with weight variance $2/n$, identity paths He et al. 2015, He et al. 2016
Optimizer fixes?	Lecture 26: let the step react to the gradient's size

Gradients are cheap, but their size is uncontrolled

Further reading (optional)

Topic	Explore further
ReLU kinks	How should automatic differentiation handle ReLU kinks? Park, Chun, Lee (ICLR 2024)
Memory	How can activation recomputation trade computation for memory? Gusak et al. (ICML 2025)
Residual scaling	How can parameterization support learning-rate transfer across widths and depths? Bordelon et al. (ICLR 2024)

Questions?

(**Email: dongshed@utk.edu**)