Lecture 21: 4/5/17

Backprop Ex#1

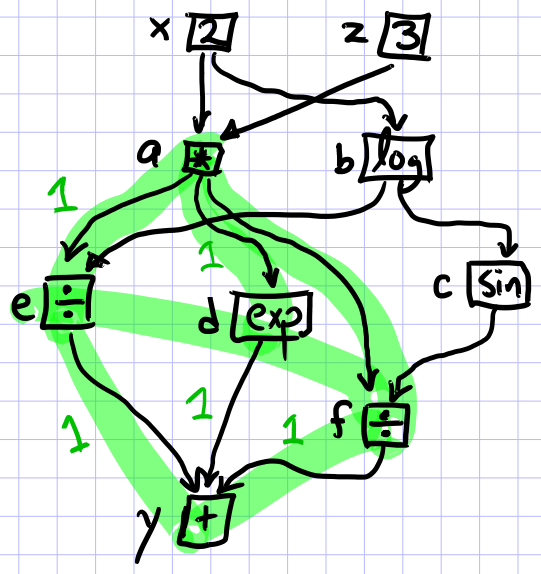
$$y = f(x,z) = \exp(xz) + \frac{xz}{\log(x)} + \frac{\sin(\log(x))}{xz}$$

Forward Computation:

Given $x = 2$, $z = 3$.

$a = xz$
$b = \log(x)$
$c = \sin(b)$
$d = \exp(a)$
$e = a/b$
$f = c/a$
$y = d + e + f$

Computation Graph:



Backward:

$\frac{dy}{dy} = 1$

$\frac{dy}{df} = 1 \quad \frac{dy}{de} = 1 \quad \frac{dy}{dd} = 1$

$\frac{dy}{dc} = \frac{dy}{df}\cdot\frac{df}{dc} = \frac{dy}{df}\left(\frac{1}{a}\right)$

$\frac{dy}{db} = \frac{dy}{de}\frac{de}{db} + \frac{dy}{dc}\frac{dc}{db}$

$= \frac{dy}{de}\left(-\frac{a}{b^2}\right) + \frac{dy}{dc}\left(\cos(b)\right)$

$\frac{dy}{da} = \frac{dy}{df}\frac{df}{da} + \frac{dy}{de}\frac{de}{da} + \frac{dy}{dd}\frac{dd}{da}$

$= \frac{dy}{df}\left(\frac{1}{b}\right) + \frac{dy}{de}\left(\exp(a)\right) + \frac{dy}{dd}\left(-\frac{c}{a^2}\right)$

$\frac{dy}{dx} = \frac{dy}{da}\frac{da}{dx} + \frac{dy}{db}\frac{db}{dx}$

$= \frac{dy}{da}(z) + \frac{dy}{db}\left(\frac{1}{x}\right)$

$\frac{dy}{dz} = \frac{dy}{da}\frac{da}{dz} = \frac{dy}{da}(x)$

Why is this efficient?

① Reuses comp. from forward in backward
② Reuses partial derivatives (but only if algorithm reuses shared computation)

* Understand partial derivatives in backward to be variables.

Neural Network Training

Consider a "3-layer" NN (matrices, vector)

- Params are $\vec{\Theta} = [\alpha^{(1)}, \alpha^{(2)}, \beta]$
- SGD Training

  Iterate until convergence:

  ① Sample $i \in \{1, \ldots, N\}$

  ② $\vec{\theta} \leftarrow \vec{\theta} - \gamma \nabla J_i(\theta)$

  where $J_i(\theta) = \ell(h_\theta(x^{(i)}), y^{(i)}) + \lambda\|\theta\|_2^2$

  (loss, output of NN, regularizer.)

- We just need $\nabla J_i(\theta) = \left[\frac{dJ_i(\theta)}{d\theta_1}, \ldots, \frac{dJ_i(\theta)}{d\theta_R}\right]^T$

Backprop Ex#2

Given: Dec. fn. $y = h_\alpha(\vec{x}) = \sigma\left((\alpha^{(3)})^T \sigma\left((\alpha^{(2)})^T \sigma\left((\alpha^{(1)})^T \vec{x}\right)\right)\right)$

Loss fn. $J = \ell(y, y^*) = y^* \log(y) + (1-y^*)\log(1-y)$

Forward:

Given $\vec{x}, \alpha^{(1)}, \alpha^{(2)}, \alpha^{(3)}$

$\vec{z}^{(0)} = \vec{x}$

for $i = 1, 2, 3$:

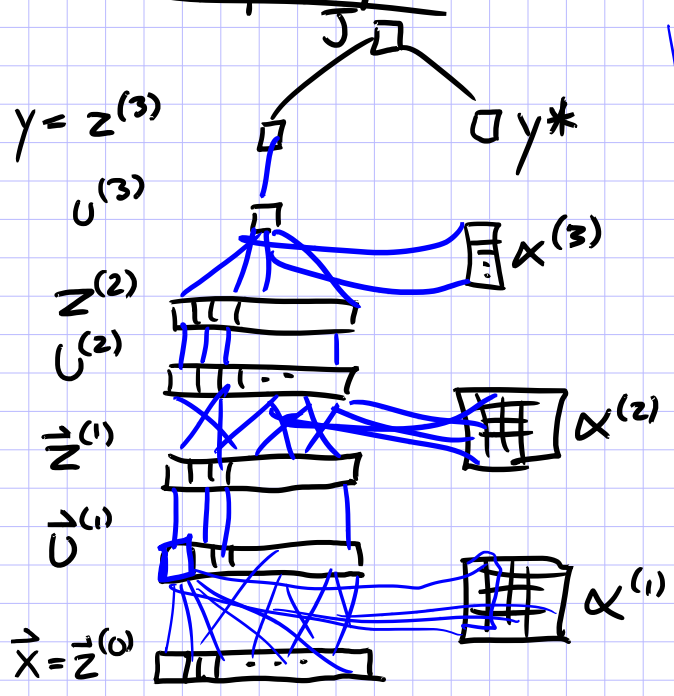
$\vec{u}^{(i)} = (\alpha^{(i)})^T \vec{z}^{(i-1)}$

$\vec{z}^{(i)} = \sigma(\vec{u}^{(i)})$

$y = z^{(3)}$

$J = \ell(y, y^*)$

Comp Graph



Backward:

Define $\nabla_{\vec{u}} \triangleq \nabla_{\vec{u}} J$

$\nabla_J = [1]$

$\nabla_y = \frac{y^*}{y} - \frac{1-y^*}{1-y}$

for $i = 3, 2, 1$:

$\nabla_{\vec{u}^{(i)}} = \nabla_{\vec{z}^{(i)}} \odot z^{(i)} \odot (1 - z^{(i)})$

$\nabla_{z^{(i-1)}} = \alpha^{(i)} \nabla_{\vec{u}^{(i)}}$

$\nabla_{\alpha^{(i)}} = \nabla_{\vec{u}^{(i)}} \otimes \vec{z}^{(i-1)}$

$\nabla_{\vec{x}} = \ldots$

SGD for CNNs

Ex: Architecture

$J = \ell(y, y^*)$

$y = \text{softmax}(z^{(5)})$

$z^{(5)} = \text{linear}(z^{(4)}, \omega)$

$z^{(4)} = \text{relu}(z^{(3)})$

$z^{(3)} = \text{conv}(z^{(2)}, \beta)$

$z^{(2)} = \text{maxpool}(z^{(1)})$

$z^{(1)} = \text{conv}(\vec{x}, \alpha)$

Parameters $\vec{\theta} = [\alpha, \beta, \omega]$

SGD:

1. Init $\vec{\theta}$
2. While not conv.

Sample $i \in \{1, \ldots, N\}$

Forward: $y = h_\theta(x^{(i)})$

$J_i(\theta) = \ell(y, y^{(i)})$

Backward:

$\nabla_{\vec{\theta}} J_i(\theta) = \ldots$

Update:

$\vec{\theta} \leftarrow \vec{\theta} - \gamma J_i(\theta)$

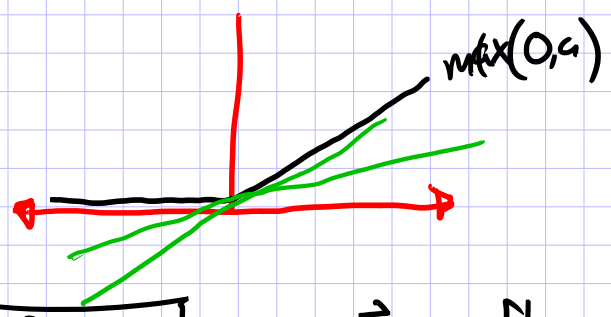
ReLU Layer

Input: $\vec{x} \in \mathbb{R}^K$ Output: $\vec{y} \in \mathbb{R}^K$

Forward:

$$\vec{y} = \sigma(\vec{x}) \leftarrow \text{element-wise}$$

$$\sigma(a) = \max(0, a)$$



Backward:

$$\frac{dJ}{dx_i} = \frac{dJ}{dy_i}\frac{dy_i}{dx_i}$$

subderivative

$$\text{where } \frac{dy_i}{dx_i} = \begin{cases} 1 & \text{if } x_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

Softmax

$$\vec{y} = \text{softmax}(\vec{x})$$

Forward:

$$y_i = \frac{\exp(x_i)}{\sum_{k=1}^{K} \exp(x_k)}$$

Back:

$$\frac{dJ}{dx_j} = \sum_{i=1}^{K} \frac{dJ}{dy_i}\frac{dy_i}{dx_j}$$

$$\text{where } \frac{dy_i}{dx_j} = \begin{cases} y_i(1-y_i) & \text{if } i=j \\ -y_i y_j & \text{otherwise} \end{cases}$$

Linear Layer

See NN part

★ trick: we "vectorize" the tensor output of previous layer