In this course we will need to learn the weights of the neural network. As we have discussed in lecture that we will be using gradient descent to learn the network weights.

$$W^{(K+1)} = W^{(K)} - \varepsilon \nabla_W \mathcal{L}$$

In the gradient descent step we will need to compute $\nabla_W \mathcal{L}$. In a neural network the weights in the earlier layers are connected to the loss function through a composition of functions, and in such cases computing the gradients require repeated application of the chain rule.

In this discussion we will go over a computational mechanism that will enable us to compute the gradients in an efficient and modular manner.

Chain rule revisited:

Let's consider the following regularized linear classification model

$$z = wx + b$$

$$y = \sigma(z)$$

$$\mathcal{L} = \frac{1}{2}(y - t)^2$$

$$R = \frac{1}{2}w^2$$

$$\mathcal{L}_{reg} = \mathcal{L} + \lambda R$$

We will learn the parameters w and b using gradient descent. Let's use chain rule to compute the partial derivatives $\frac{\partial \mathcal{L}_{reg}}{\partial w}$, $\frac{\partial \mathcal{L}_{reg}}{\partial b}$

$$\mathcal{L}_{reg} = \frac{1}{2}\left[\sigma(wx+b) - t\right]^2 + \frac{\lambda}{2} w^2$$

$$\frac{\partial \mathcal{L}_{reg}}{\partial w} = \frac{1}{2}\frac{\partial}{\partial w}\left[\sigma(wx+b)-t\right]^2 + \frac{\lambda}{2}\frac{\partial}{\partial w} w^2$$

$$= \left[\sigma(wx+b)-t\right]\frac{\partial}{\partial w}\left[\sigma(wx+b)-t\right] + \lambda w$$

$$= \left[\sigma(wx+b)-t\right]\sigma'(wx+b)\frac{\partial}{\partial w}(wx+b) + \lambda w$$

$$= \left[\sigma(wx+b)-t\right]\sigma'(wx+b)\,x + \lambda w$$

$$\frac{\partial \ell_{reg}}{\partial b} = \frac{\partial}{\partial b}\left[\frac{1}{2}(\sigma(wx+b)-t)^2 + \frac{\lambda}{2}w^2\right]$$

$$= \frac{1}{2}\frac{\partial}{\partial b}\left[\sigma(wx+b)-t\right]^2 + \frac{\lambda}{2}\frac{\partial}{\partial b}w^2$$

$$= \left[\sigma(wx+b)-t\right]\frac{\partial}{\partial b}\left[\sigma(wx+b)-t\right] + 0$$

$$= \left[\sigma(wx+b)-t\right]\sigma'(wx+b)\frac{\partial}{\partial b}(wx+b)$$

$$= \left[\sigma(wx+b)-t\right]\sigma'(wx+b)$$

The above computation involves

(a) Lots of copies of terms from one line to next

(b) Lots of redundant work. For instance, the first 3 steps in the two derivations above are nearly identical.

The idea behind backprop is to share repeated computations wherever possible.

Computational graphs and backprop:

Computational graphs help to visualize and contextualize how we aim to backpropagate derivatives. It is the mechanism by which deeplearning libraries compute gradients.

A computational graph is a directed acyclic graph where each node in the graph denote a mathematical operation.

Let's draw the computational graph for our running example

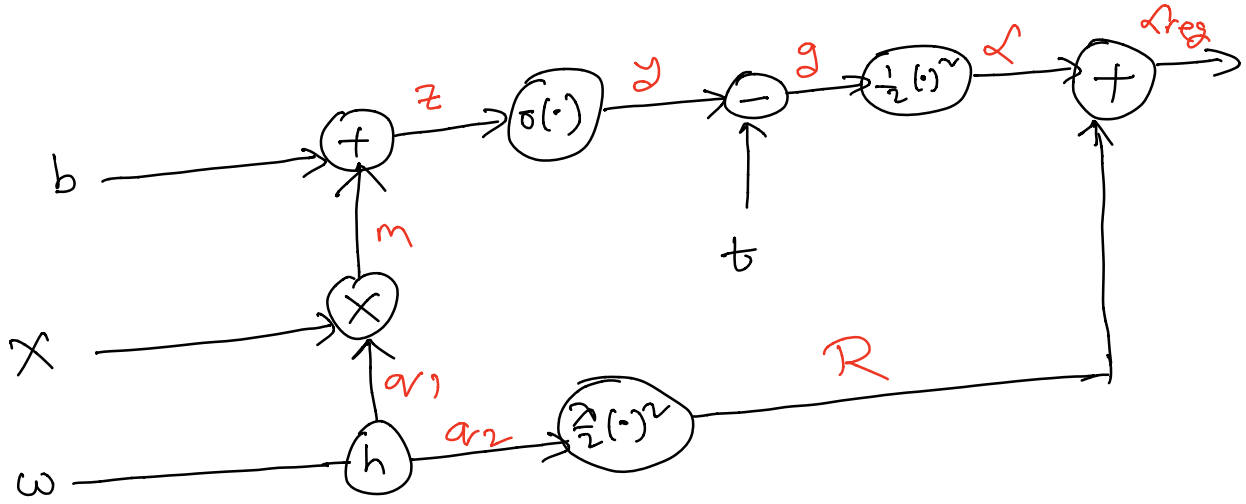
$$z = wx + b$$

$$y = \sigma(z)$$

$$\mathcal{L} = \frac{1}{2}(y - t)^2$$

$$R = \frac{1}{2}w^2$$

$$\mathcal{L}_{reg} = \mathcal{L} + \lambda R$$



The backprop algorithm consists of two steps:

(i) In forward pass we take the input X and the initial values of W and b and propagate in the forward direction through the network to compute the quantities in red.

(ii) In backward pass we compute the derivatives starting at the output and propagating it backward to the input.

So,

$$\mathcal{L}_{reg} = \mathcal{L} + R$$

$$\frac{\partial \mathcal{L}_{reg}}{\partial \mathcal{L}} = 1, \quad \frac{\partial \mathcal{L}_{reg}}{\partial R} = 1$$

$$\mathcal{L} = \frac{1}{2} g^2$$

$$\frac{\partial \mathcal{L}}{\partial g} = g$$

$$\frac{\partial \mathcal{L}_{reg}}{\partial g} = \frac{\partial \mathcal{L}}{\partial g} \frac{\partial \mathcal{L}_{reg}}{\partial \mathcal{L}}$$

$$= g$$

$$R = \frac{\lambda}{2} a_2^2$$

$$\frac{dR}{da_2} = \lambda a_2$$

$$\frac{d\mathcal{L}_{reg}}{da_2} = \frac{dR}{da_2} \frac{d\mathcal{L}_{reg}}{dR}$$

$$= \lambda a_2$$

$$g = y - t$$

$$\frac{\partial g}{\partial y} = 1$$

$$\frac{\partial L_{reg}}{\partial y} = \frac{\partial s}{\partial y} \cdot \frac{\partial L_{reg}}{\partial s}$$

$$= g$$

$$y = \sigma(z)$$

$$\frac{\partial y}{\partial z} = \sigma'(z)$$

$$\frac{\partial L_{reg}}{\partial z} = \frac{\partial y}{\partial z} \cdot \frac{\partial L_{reg}}{\partial y}$$

$$= \sigma'(z)\, g$$

$$z = b + m$$

$$\frac{\partial z}{\partial b} = 1, \quad \frac{\partial z}{\partial m} = 1$$

$$\frac{\partial L_{reg}}{\partial b} = \frac{\partial z}{\partial b} \cdot \frac{\partial L_{reg}}{\partial z}$$

$$= \sigma'(z)\, g = \sigma'(wx+b)\left[\sigma(wx+b) - t\right]$$

$$\frac{\partial \mathcal{L}_{reg}}{\partial m} = \frac{\partial z}{\partial m} \frac{\partial \mathcal{L}_{reg}}{\partial z}$$

$$= \sigma'(z)\, g$$

$$m = q_1 x$$

$$\frac{\partial m}{\partial q_1} = x \qquad \frac{\partial m}{\partial x} = q_1$$

$$\frac{\partial \mathcal{L}_{reg}}{\partial x} = \frac{\partial m}{\partial x} \frac{\partial \mathcal{L}_{reg}}{\partial m}$$

$$= q_1 \sigma'(z)\, g$$

$$\frac{\partial \mathcal{L}_{reg}}{\partial q_1} = \frac{\partial m}{\partial q_1} \frac{\partial \mathcal{L}_{reg}}{\partial m}$$

$$= x\, \sigma'(z)\, g$$

Now by law of total derivatives,

$$\frac{\partial L_{reg}}{\partial w} = \frac{\partial a_1}{\partial w}\frac{\partial L_{reg}}{\partial a_1} + \frac{\partial a_2}{\partial w}\frac{\partial L_{reg}}{\partial a_2}$$

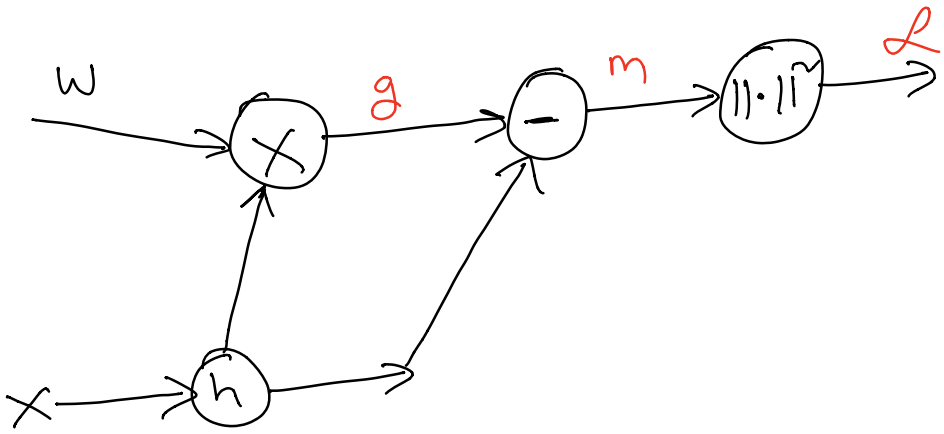
$$\frac{\partial L_{reg}}{\partial w} = \frac{\partial L_{reg}}{\partial a_1} + \frac{\partial L_{reg}}{\partial a_2}$$

$$= x\sigma'(z)g + \lambda a_2$$

$$= x\sigma'(wx+b)\left[\sigma(wx+b) - t\right] + \lambda w$$

Practice problems:

1



In the above computational graph,

$$h(x) = Ix; \quad I \in \mathbb{R}^{n \times n}$$

$$g = Wx; \quad g \in \mathbb{R}^n$$

$$m = g - x; \quad m \in \mathbb{R}^n$$

So,

$$\mathcal{L} = \|m\|_2^2$$

$$\nabla_m \mathcal{L} = 2m \in \mathbb{R}^n \quad (\text{Eqn } 13)$$

$$m = g - x$$

$$\nabla_g m = I \in \mathbb{R}^{n \times n}$$

$$\nabla_g \ell = \nabla_g m \; \nabla_m \ell$$

$$= 2m \in \mathbb{R}^n$$

$$g = wx$$

$$\nabla_w g = \nabla_w (wx) \in \mathbb{R}^{n \times n \times m}$$

$$\nabla_w \ell = \nabla_w g \; \nabla_g \ell$$

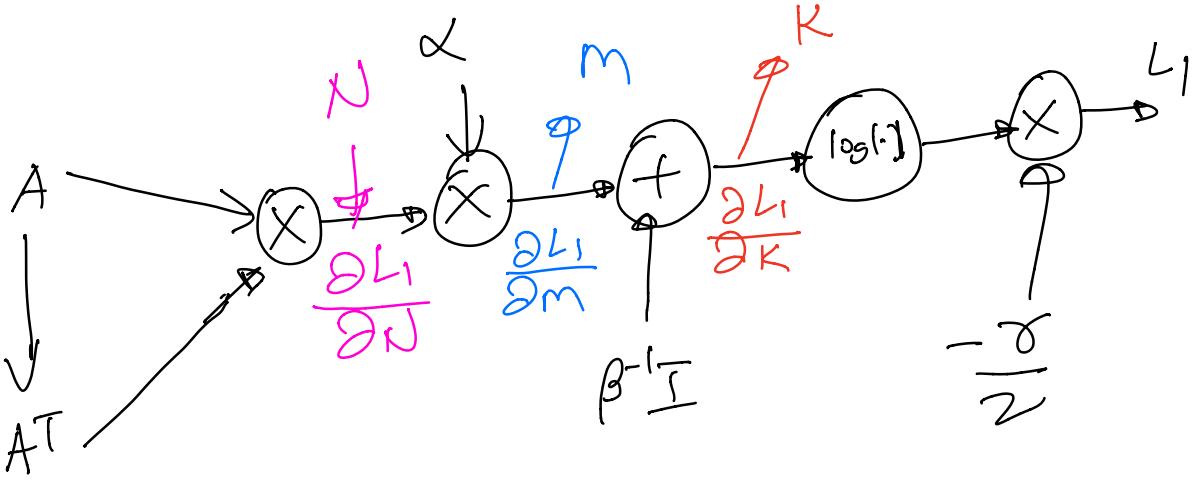
$$= \nabla_g \ell \; x^T$$

$$= 2(wx - x) x^T$$

$$\therefore \nabla_w \ell = 2(wx - x) x^T$$

2

a) The computational graph for L_1 is drawn below:



From the computational graph,

$$L_1 = -\frac{\gamma}{2} \log|K| = -\frac{\gamma}{2} \log(\det K)$$

Using matrix cookbook

$$\frac{\partial L_1}{\partial K} = -\frac{\gamma}{2} (K^{-1})^T = -\frac{\gamma}{2} (K^T)^{-1}$$

(Eqn 57)

Since $(+)$ operator distributes the gradient, so

$$\frac{\partial L_1}{\partial m} = -\frac{\gamma}{2}(K^T)^{-1}$$

Since $(\times)$ operator switches the gradient, so

$$\frac{\partial L_1}{\partial N} = -\frac{\alpha\gamma}{2}(K^T)^{-1}$$

Now,

$$N = AA^T$$

using the hint given in the problem

$$\frac{\partial L_1}{\partial A} = \frac{\partial L_1}{\partial N}(A^T)^T$$

$$\frac{\partial L_1}{\partial A} = -\frac{\alpha\gamma}{2}(K^T)^{-1}A$$

Since A^T has the same elements as A, so taking the gradient of L_1 w.r.t A^T we have

$$\frac{\partial L_1}{\partial A^T} = A^T \frac{\partial L_1}{\partial N}$$

$$= A^T \left[-\frac{\alpha\gamma}{2} (K^T)^{-1} \right]$$

$$\frac{\partial L_1}{\partial A^T} = \frac{-\alpha\gamma}{2} (K^{-1}A)^T$$

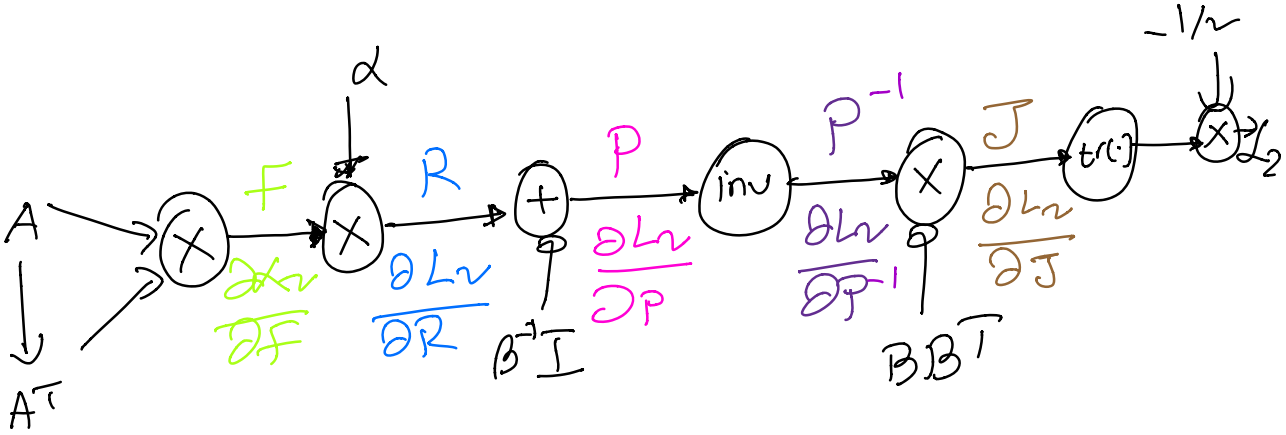
Now adding the gradient paths through A and A^T we get

$$\frac{\partial L_1}{\partial A} = \frac{-\alpha\gamma}{2} (K^T)^{-1} A - \left[\frac{\alpha\gamma}{2} (K^{-1}A)^T \right]^T$$

Since K is symmetric,

$$\frac{\partial L_1}{\partial A} = -\alpha\gamma K^{-1} A$$

b) The computational graph for L_2 is drawn below:



From the computational graph,

$$L_2 = -\frac{1}{2} tr(J)$$

Now, from matrix cookbook

$$\frac{\partial L_2}{\partial J} = -\frac{1}{2} I \quad (\text{Eqn } 99)$$

From computational graph,

$$J = P^{-1}(BB^T)$$

Hence using the hint

$$\frac{\partial L_N}{\partial P^{-1}} = \frac{-1}{2} BB^T$$

Now using the given hint

$$\frac{\partial L_N}{\partial P} = -P^{-1} \frac{\partial L_2}{\partial P^{-1}} P^{-1}$$

Since (+) operator distributes the gradient, so

$$\frac{\partial L_N}{\partial R} = -P^{-1} \frac{\partial L_N}{\partial P^{-1}} P^{-1}$$

Since ⊗ operator switches the gradient,

so

$$\frac{\partial \mathcal{L}_N}{\partial F} = -\alpha P^{-1} \frac{\partial \mathcal{L}_N}{\partial P^{-1}} P^{-1}$$

Now,

$$F = AA^T$$

using the given hint,

$$\frac{\partial \mathcal{L}_N}{\partial A} = -\alpha P^{-1} \frac{\partial \mathcal{L}_N}{\partial P^{-1}} P^{-1} A$$

$$\frac{\partial \mathcal{L}_N}{\partial A} = \frac{+\alpha}{2} P^{-1} BB^T P^{-1} A$$

Using the same reasoning as (a) to compute the derivative of L_2 with respect to A^T

$$\frac{\partial L_2}{\partial A^T} = -\alpha A^T P^{-1} \frac{\partial \alpha_2}{\partial P^{-1}} P^{-1}$$

$$= \frac{\alpha}{2} A^T P^{-1} B B^T P^{-1}$$

Now adding the gradients paths through A and A^T we get:

$$\frac{\partial L_2}{\partial A} + \left[\frac{\partial L_2}{\partial A^T}\right]^T$$

$$= \frac{\alpha}{2} P^{-1} B B^T P^{-1} A + \left[\frac{\alpha}{2} A^T P^{-1} B B^T P^{-1}\right]^T$$

$$= \frac{\alpha}{2} P^{-1} B B^T P^{-1} A + \frac{\alpha}{2} (P^T)^{-1} B B^T (P^T)^{-1} A$$

Since P is symmetric so

$$= \frac{\alpha}{2} P^{-1} B B^T P^{-1} A + \frac{\alpha}{2} P^{-1} B B^T P^{-1} A$$

$$= \alpha P^{-1} B B^T P^{-1} A$$

3

We will use backpropagation to compute the required gradients.

Starting from the loss function L

$$L = \|q\|_2^2$$

$$\frac{dL}{dq} = 2q \in \mathbb{R}^K$$

Now, $q = p - y$

Computing the local gradient at $\ominus$

$$\frac{dq}{dp} = I \in \mathbb{R}^{K \times K}$$

$$\frac{dq}{dy} = -I \in \mathbb{R}^{K \times K}$$

Using chain rule, backpropagating to p, we have

$$\frac{\partial L}{\partial p} = \frac{\partial q}{\partial p} \frac{\partial L}{\partial q}$$

$$\frac{\partial L}{\partial p} = 2q \in \mathbb{R}^K$$

Now,

$$p = 2^{z_2}$$

where we raise each entry of z_2 to the power of 2

$$2^{z_2} = \begin{bmatrix} 2^{z_{2,1}} \\ 2^{z_{2,2}} \\ 2^{\cdots} \\ \vdots \\ 2^{z_{2,K}} \end{bmatrix} \in \mathbb{R}^K$$

Computing the local gradient @ $(2^{(\cdot)})$:

$$\frac{dP}{dz_2} = \text{diag}\left(2^{z_2} \ln 2\right) \in \mathbb{R}^{K \times K}$$

Using chain rule backpropagating to z_2, we have

$$\frac{dL}{dz_2} = \frac{dP}{dz_2} \frac{dL}{dP}$$

$$\frac{dL}{dz_2} = 2^{z_2} \ln 2 \odot 2a \in \mathbb{R}^K$$

since ⊕ gate passes the gradient, so

$$\frac{dL}{du} = 2^{z_2} \ln 2 \odot 2q \in \mathbb{R}^K$$

$$\frac{dL}{db_2} = 2^{z_2} \ln 2 \odot 2q \in \mathbb{R}^K$$

Now,

$$u = W_2 h_1$$

Computing the local gradient @

⊗

$$\frac{du}{dh_1} = W_2^T \in \mathbb{R}^{m \times K}$$

Using chain rule, backpropagating to h_1 we have

$$\frac{dL}{dh_1} = \frac{du}{dh_1} \frac{dL}{du}$$

$$\frac{dL}{dh_1} = W_2^T \frac{dL}{du} \in \mathbb{R}^m$$

By the 3-D tensor derivative trick learned in class,

$$\frac{dL}{dW_2} = \frac{dL}{du} h_1^T \in \mathbb{R}^{K \times m}$$

Now, $h_1 = \text{relu}(z_1)$

Computing the local gradient @ relu:

$$\frac{dh_1}{dz_1} = \text{diag}(z_1 > 0) \in \mathbb{R}^{m \times m}$$

using chain rule, backpropagating to z_1 we have

$$\frac{dL}{dz_1} = \frac{dh_1}{dz_1} \frac{dL}{dh_1}$$

$$= \text{diag}(z_1 > 0) \frac{dL}{dh_1}$$

$$\frac{dL}{dz_1} = \mathbb{I}(z_1 > 0) \odot \frac{dL}{dh_1} \in \mathbb{R}^m$$

since $\oplus$ passes the gradient, so

$$\frac{dL}{db_1} = \frac{dL}{dz_1} \in \mathbb{R}^m$$

$$\frac{dL}{dt} = \frac{dL}{dz_1} \in \mathbb{R}^m$$

Now,

$$t = W_1 X$$

Computing the local gradient @ (X)

$$\frac{\partial t}{\partial X} = W_1^T \in \mathbb{R}^{n \times m}$$

using chain rule, backpropagating to X we have

$$\frac{\partial L}{\partial X} = \frac{\partial t}{\partial X} \frac{\partial L}{\partial t}$$

$$\frac{\partial L}{\partial X} = W_1^T \frac{\partial L}{\partial Z_1} \in \mathbb{R}^n$$

By the 3-D tensor derivative trick learned in class,

$$\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial t} X^T$$

$$\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial Z_1} X^T \in \mathbb{R}^{m \times n}$$