RECITATION 5: NEURAL NETS

10-301/10-601 Introduction to Machine Learning (Summer 2024)
https://www.cs.cmu.edu/~hchai2/courses/10601/

1 Matrix Calculus

Consider $\mathbf{x} \in \mathbb{R}^p$, $\mathbf{y} \in \mathbb{R}^r$, $\mathbf{z} \in \mathbb{R}^n$ where $\mathbf{z} = g(\mathbf{y})$, and $\mathbf{y} = f(\mathbf{x})$. We want to derive $d\mathbf{z}/d\mathbf{x}$ (a vector form of the scalar chain rule).

1. If $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ were all scalars, what would dz/dx be?

   Solution

   $$\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx}$$

   Shape matching:

2. Fill in the following shapes:

   $$\frac{d\mathbf{y}}{d\mathbf{x}}: \qquad \frac{d\mathbf{z}}{d\mathbf{y}}: \qquad \frac{d\mathbf{z}}{d\mathbf{x}}:$$

   Solution

   $$\frac{d\mathbf{y}}{d\mathbf{x}}: p \times r \quad \frac{d\mathbf{z}}{d\mathbf{y}}: r \times n \quad \frac{d\mathbf{z}}{d\mathbf{x}}: p \times n$$

3. Therefore, the correct derivative is

   $$\frac{d\mathbf{z}}{d\mathbf{x}} =$$

   Solution

   $$\frac{d\mathbf{z}}{d\mathbf{x}} = \frac{d\mathbf{y}}{d\mathbf{x}}\frac{d\mathbf{z}}{d\mathbf{y}}$$

   Generalizing a single element: The more rigorous method of computing such derivatives is by computing the scalar derivative for a single element, then generalizing this to all elements by turning the

scalar form into a matrix form.

4. Fill in the single element:

$$\frac{dz_k}{dx_i} =$$

Solution

$$\frac{dz_k}{dx_i} = \sum_j \frac{dz_k}{dy_j}\frac{dy_j}{dx_i} = \sum_j \frac{dy_j}{dx_i}\frac{dz_k}{dy_j}$$

2 Forward Propagation

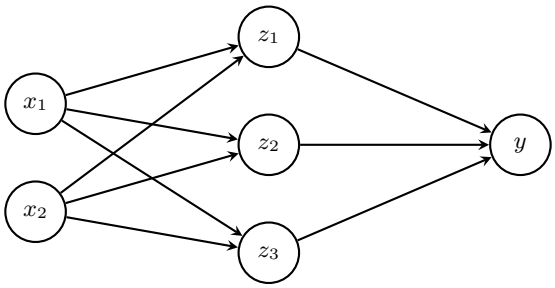


Figure 1: Neural Network For Example Questions

Forward Propagation is the process of calculating the value of your loss function, given data, weights and activation functions. Given the input data $\mathbf{x}$, we can transform it by the given weights, $\boldsymbol{\alpha}$, then apply the corresponding activation function to it and finally pass the result to the next layer. Forward propagation does not involve taking derivatives and proceeds from the input layer to the output layer.

Network Overview Consider the neural network with one hidden layer shown in Figure 2. The input layer consists of 2 features $\mathbf{x} = [x_1, x_2]^T$, the hidden layer has 3 nodes with output $\mathbf{z} = [z_1, z_2, z_3]^T$, and the output layer is a scalar $\hat{y}$. We also add a bias to the input, $x_0 = 1$ and the output of the hidden layer $z_0 = 1$, both of which are fixed to 1.

$\boldsymbol{\alpha}$ is the matrix of weights from the inputs to the hidden layer and $\boldsymbol{\beta}$ is the matrix of weights from the hidden layer to the output layer. $\alpha_{j,i}$ represents the weight going *to* the node z_j in the hidden layer *from* the node x_i in the input layer (e.g. $\alpha_{1,2}$ is the weight from x_2 to z_1), and $\boldsymbol{\beta}$ is defined similarly. We will use a **tanh** activation function for the hidden layer and no activation for the output layer.

Network Details Equivalently, we define each of the following.

The input:

$$\mathbf{x} = [x_0, x_1, x_2]^T \tag{1}$$

Linear combination at the first (hidden) layer:

$$a_j = \sum_{i=0}^{2} \alpha_{j,i} \cdot x_i, \ \forall j \in \{1, \ldots, 3\} \tag{2}$$

Activation at the first (hidden) layer:

$$z_j = \tanh(a_j) = \frac{e^{a_j} - e^{-a_j}}{e^{a_j} + e^{-a_j}}, \ \forall j \in \{1, \ldots, 3\} \tag{3}$$

$$\mathbf{z} = [z_0, z_1, z_2, z_3]^T \tag{4}$$

Linear combination at the second (output) layer:

$$\hat{y} = \sum_{j=0}^{3} \beta_j \cdot z_j, \tag{5}$$

Here we fold in the bias term $\alpha_{j,0}$ by thinking of $x_0 = 1$, and fold in β_0 by thinking of $z_0 = 1$.

Loss We will use Squared error loss, $\ell(\hat{y}, y)$:

$$\ell(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2 \tag{6}$$

We initialize the network weights as:

$$\boldsymbol{\alpha} = \begin{bmatrix} 0 & 1 & 2 \\ 2 & 1 & 0 \\ 0 & 2 & 0 \end{bmatrix}$$

$$\boldsymbol{\beta} = \begin{bmatrix} 0 & 1 & 2 & 2 \end{bmatrix}$$

For the following questions, we use $y = 3$.

1. Why and how do we include a bias term in the input and in the hidden-layer?

   Solution Similar to how an intercept term in linear regression allows it to better fit data, the bias term helps the neural network better fit its data as well.

   We simply fold in the bias term into our input vector as the 0th term and make the value equals 1.

2. Why do we need to use nonlinear activation functions in our neural net?

   Solution A neural network with only linear activation functions would be no different than a linear regression. (Try forward propagating with only linear functions on the given example)

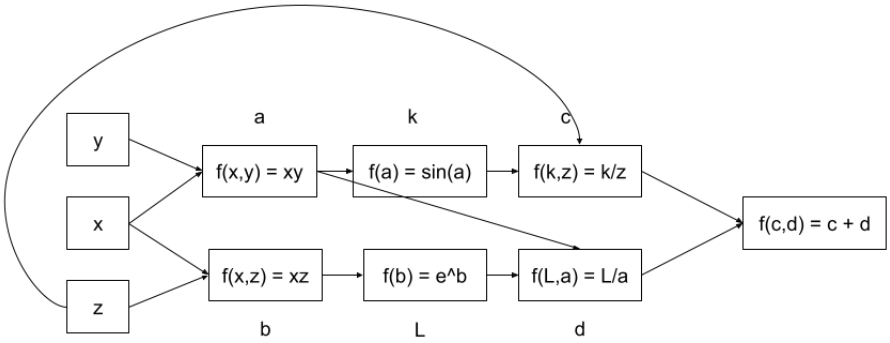
3. **Vector Form:** Find the vector form of forward computation, given $\mathbf{x}$ is a column vector. **Solution**

$$
\begin{aligned}
\mathbf{a} &= \boldsymbol{\alpha}\hat{\mathbf{x}} \\
&= \begin{bmatrix} \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} \\ \alpha_{3,0} & \alpha_{3,1} & \alpha_{3,2} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} \\
&= \begin{bmatrix} \alpha_{1,0}x_0 + \alpha_{1,1}x_1 + \alpha_{1,2}x_2 \\ \alpha_{2,0}x_0 + \alpha_{2,1}x_1 + \alpha_{2,2}x_2 \\ \alpha_{3,0}x_0 + \alpha_{3,1}x_1 + \alpha_{3,2}x_2 \end{bmatrix} \\
&= \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \qquad (7) \\
\mathbf{z} &= \tanh(\mathbf{a}) \\
\hat{y} &= \boldsymbol{\beta}\hat{\mathbf{z}} \\
&= \begin{bmatrix} \beta_0 & \beta_1 & \beta_2 & \beta_3 \end{bmatrix} \begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{bmatrix} \\
&= \beta_0 z_0 + \beta_1 z_1 + \beta_2 z_2 + \beta_3 z_3
\end{aligned}
$$

3 Diagrams

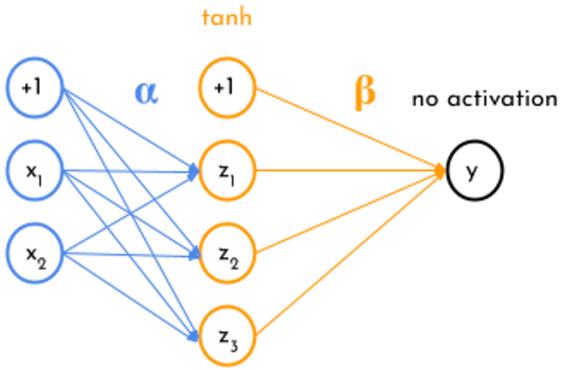
1. For the following function f, create the computational graph using the conventions defined in lecture.

$$f(x, y, z) = \frac{\sin(xy)}{z} + \frac{e^{xz}}{xy}$$

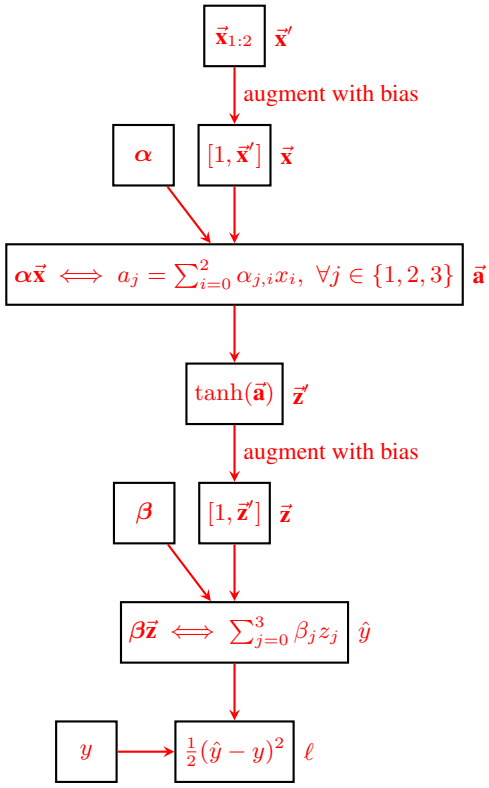


Solution

2. For the following neural network, draw the corresponding computational graph. Assume that all hidden units use the tanh function as the activation function and that the loss is mean squared error. Provide the shape of all parameters defined in the computational graph. Assume the weights for the first layer and second layers are respectively the matrices α and β.



Solution



$$\boldsymbol{\alpha} \in \mathbb{R}^{3\times 3} \quad \boldsymbol{\beta} \in \mathbb{R}^4$$

4 Backward Propagation

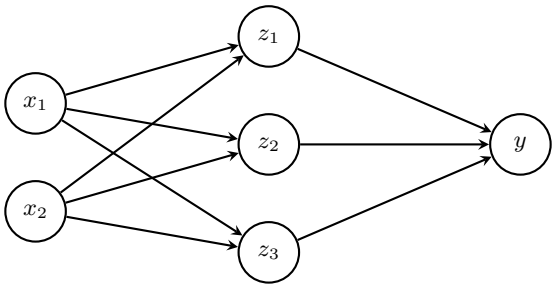


Figure 2: Neural Network For Example Questions (same as Figure 1)

Given a Neural Network and a corresponding loss function $J(\theta)$, backpropagation gives us the gradient of the loss function with respect to the weights of the neural network. The method is called *backward* propagation because we calculate the gradients of the final layer of weights first, then proceed backward to the first layer. In a simple neural network with one hidden layer, the partial derivatives that we need for learning are $\frac{\partial \ell}{\partial \alpha_{ij}}$ and $\frac{\partial \ell}{\partial \beta_{kj}}$, and we need to apply chain rule recursively to obtain these. Note that in implementation, it is easier to use matrix/vector forms to conduct computations.

1. Many gradients are calculated in back propagation. Which of these gradients are used to update the weights? Do not include intermediate value(s) used to calculate these gradient(s).
   Solution The gradients with respect to α and β are used in updating. The rest are intermediate values used to calculate these two gradients

2. **Scalar Form:** Given $x_1 = 1$, $x_2 = 2$, what are the values of $\frac{\partial \ell}{\partial \beta_1}$, $\frac{\partial \ell}{\partial \alpha_{1,1}}$?

   Hint: Derive expressions for $\frac{\partial \ell}{\partial \beta_i}$ and $\frac{\partial \ell}{\partial \alpha_{i,j}}$ first, then substitute in values.

   For convenience, the computation graph for the neural network is displayed below:

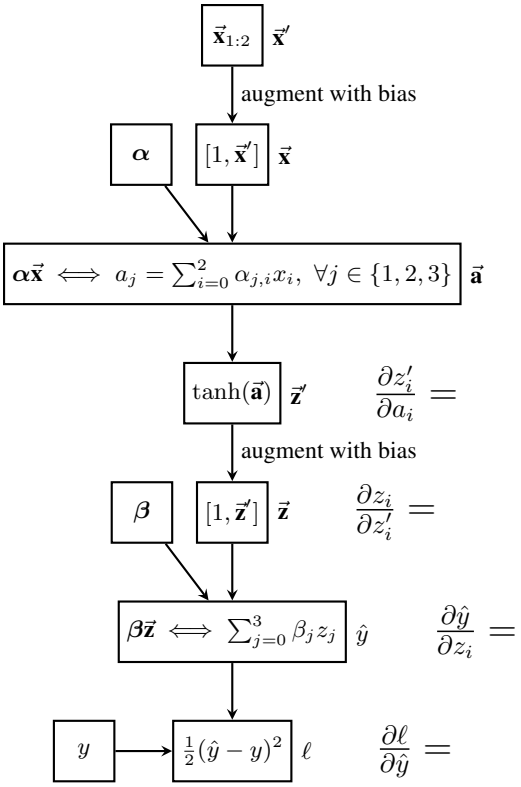


Figure 3: Computation graph for the neural network in Figure 2

Hint: $\frac{\partial \tanh(x)}{\partial x} = 1 - \tanh(x)^2$

Table 1: $tanh$ values

x	1	2	3	4	5	6	7	8	9
$\tanh(x)$	0.76159	0.96403	0.99505	0.99933	0.99991	0.99999	0.99999	0.99999	0.99999

$$\frac{\partial \ell}{\partial \beta_i} = \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \beta_i} \qquad \frac{\partial \ell}{\partial \beta_1} =$$

Solution As a reminder, we were given that $\ell = \frac{1}{2}(\hat{y} - y)^2$ and $\hat{y} = \sum_{j=0}^{3} \beta_j z_j$

So we can calculate:

$$\begin{aligned}
\frac{\partial \ell}{\partial \beta_1} &= \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \beta_1} \\
&= \frac{\partial}{\partial \hat{y}} \left[\frac{1}{2}(\hat{y} - y)^2 \right] \frac{\partial}{\partial \beta_1} \left[\sum_{j=0}^{3} \beta_j z_j \right] \\
&= (\hat{y} - y)\, z_1 = (4.91807 - 3) * 0.99991 = 1.9179
\end{aligned}$$

$$\frac{\partial \ell}{\partial \alpha_{i,j}} = \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_i} \frac{\partial z_i}{\partial a_i} \frac{\partial a_i}{\partial \alpha_{i,j}} \qquad \frac{\partial \ell}{\partial \alpha_{1,1}} =$$

Solution

$$\begin{aligned}
\frac{\partial \ell}{\partial \alpha_{1,1}} &= \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_1} \frac{\partial z_1}{\partial a_1} \frac{\partial a_1}{\partial \alpha_{1,1}} \\
&= (\hat{y} - y)\beta_1(1 - \tanh(a_1)^2)x_1 \\
&= (4.91807 - 3) * 1 * (1 - \tanh(5)^2) * 1 \\
&= 0.000348
\end{aligned}$$

3. **Vector Form:** What are the values of $\frac{\partial \ell}{\partial \boldsymbol{\beta}}$, $\frac{\partial \ell}{\partial \boldsymbol{\alpha}}$ given $x_1 = 1$, $x_2 = 2$, $y = 3$, and the network weights from the forward propagation section?

   Hint: $\frac{\partial \tanh(x)}{\partial x} = 1 - \tanh(x)^2$

   Solution

Table 2: *tanh* values

x	1	2	3	4	5	6	7	8	9
$\tanh(x)$	0.76159	0.96403	0.99505	0.99933	0.99991	0.99999	0.99999	0.99999	0.99999

$$a = \alpha \hat{x}$$

$$a = \begin{bmatrix} 5 \\ 3 \\ 2 \end{bmatrix}$$

$$z = \tanh(a)$$

$$z = \begin{bmatrix} 1 & 0.99991 & 0.99505 & 0.96403 \end{bmatrix}$$

$$\hat{y} = \beta z$$

$$\hat{y} = 4.91807$$

$$\frac{\partial \ell}{\partial \boldsymbol{\beta}} = \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \boldsymbol{\beta}} = (\hat{y} - y)\mathbf{z}^T = \begin{bmatrix} 1.91807 & 1.9179 & 1.90857 & 1.84908 \end{bmatrix}$$

Denote $\hat{\boldsymbol{\beta}}$ as $\boldsymbol{\beta}$ without the first entry.

$$\frac{\partial \hat{y}}{\partial z_i} = \boldsymbol{\beta}_i \quad \forall i \in \{1, 2, 3\}$$

$$\frac{\partial \hat{y}}{\partial \mathbf{z}} = \hat{\boldsymbol{\beta}}^T$$

$$\frac{\partial z_i}{\partial a_j} = \begin{cases} 1 - \tanh^2(a_i) & \text{when } i = j \\ 0 & \text{when } i \neq j \end{cases} \quad \forall i, j \in \{1, 2, 3\}$$

$$
\begin{aligned}
\frac{\partial \ell}{\partial \mathbf{a}} &= \frac{\partial \mathbf{z}}{\partial \mathbf{a}} \frac{\partial \hat{y}}{\partial \mathbf{z}} \frac{\partial \ell}{\partial \hat{y}} \\
&= \operatorname{diag}(1 - \tanh^2(\mathbf{a})) \cdot \hat{\boldsymbol{\beta}}^T \cdot (\hat{y} - y) \\
&= (1 - \mathbf{z}^2) \odot \hat{\boldsymbol{\beta}} \cdot (\hat{y} - y) \qquad (\odot \text{ is element-wise multiplication}) \\
&= \begin{bmatrix} 1 - \tanh^2(a_1) & 0 & 0 \\ 0 & 1 - \tanh^2(a_2) & 0 \\ 0 & 0 & 1 - \tanh^2(a_3) \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix} \cdot 1.91807 \\
&= \begin{bmatrix} 0.000182 & 0 & 0 \\ 0 & 0.009866 & 0 \\ 0 & 0 & 0.070650 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix} \cdot 1.91807 \\
&= \begin{bmatrix} 0.000348 \\ 0.037848 \\ 0.271027 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \ell}{\partial \alpha_{ji}} &= \frac{\partial \ell}{\partial a_j} \frac{\partial a_j}{\partial \alpha_{ji}} \\
&= \frac{\partial \ell}{\partial a_j} x_i
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \ell}{\partial \boldsymbol{\alpha}} &= \begin{bmatrix} \frac{\partial \ell}{\partial a_1} x_0 & \frac{\partial \ell}{\partial a_1} x_1 & \frac{\partial \ell}{\partial a_1} x_2 \\ \frac{\partial \ell}{\partial a_2} x_0 & \frac{\partial \ell}{\partial a_2} x_1 & \frac{\partial \ell}{\partial a_2} x_2 \\ \frac{\partial \ell}{\partial a_3} x_0 & \frac{\partial \ell}{\partial a_3} x_1 & \frac{\partial \ell}{\partial a_3} x_2 \end{bmatrix} \\
&= \frac{\partial \ell}{\partial \mathbf{a}} \mathbf{x}^T \\
&= \begin{bmatrix} 0.00034829 & 0.00034829 & 0.00069658 \\ 0.03784758 & 0.03784758 & 0.07569515 \\ 0.271027 & 0.271027 & 0.542054 \end{bmatrix}
\end{aligned}
$$