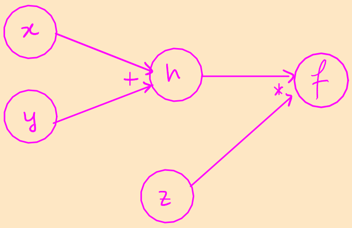
2 Backrop in Practice: Staged Computation

For the function $f(x, y, z) = (x + y)z$:

(a) Decompose f into two simpler functions.

computation graph

(b) Draw the network that represents the computation of f.



$f(x,y,z) = g(h(x,y), z)$

$g(a,b) = a \cdot b$

$h(a,b) = a+b$

(c) Write the forward pass and backward pass (backpropagation) in the network.

```
def forward(x, y, z):
    h = x + y
    f = h * z
    return f

def backward(x, y, z):
    df_dz = h
    df_dh = z
    df_dx = df_dh
    df_dy = df_dh
```

$\frac{\partial}{\partial x}(x+y) = 1$

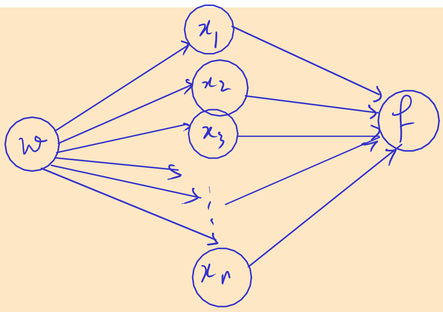
$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial h} \cdot \frac{\partial h}{\partial x} = \frac{\partial f}{\partial h}$

(d) Update your network drawing with the intermediate values in the forward and backward pass. Use the inputs $x = -2$, $y = 5$, and $z = -4$.

3 Backpropagation Practice

scalar output

(a) Chain rule of multiple variables: Assume that you have a function given by $f(x_1, x_2, \ldots, x_n)$, and that $g_i(w) = x_i$ for a scalar variable w. How would you compute $\frac{d}{dw} f(g_1(w), g_2(w), \ldots, g_n(w))$? What is its computation graph?



$$\frac{df}{dx} \cdot \frac{dx}{dw}$$

$$\sum_{i=1}^{n} \frac{\partial f}{\partial x_i} \cdot \frac{\partial x_i}{\partial w}$$

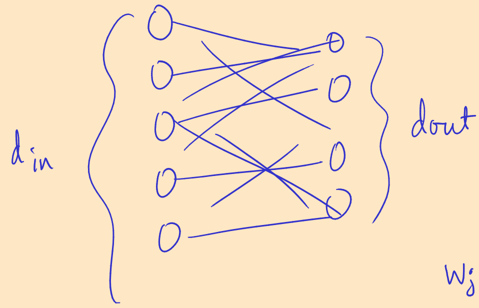
$$\| $$

$$\boxed{\underbrace{\frac{\partial f}{\partial x}}_{\text{row vec}} \circ \underbrace{\frac{\partial x}{\partial w}}_{\text{col vec}}}$$

d_n = batch size

(b) Let $Z = XW + \mathbf{1}b^\top$, where $Z \in \mathbb{R}^{d_n \times d_{out}}$, $X \in \mathbb{R}^{d_n \times d_{in}}$, $W \in \mathbb{R}^{d_{in} \times d_{out}}$, b is a ~~row~~ vector in $\mathbb{R}^{d_{out}}$, and $\mathbf{1}$ is a column vector in $\mathbb{R}^{d_n}$. Given $\frac{\partial L}{\partial Z} \in \mathbb{R}^{d_n \times d_{out}}$, where l is a scalar loss, calculate $\frac{\partial L}{\partial W}$ and $\frac{\partial L}{\partial b}$.

scalar



$$\begin{bmatrix} — x_1^T — \\ — x_2^T — \\ \vdots \\ — x_{d_n}^T — \end{bmatrix} \begin{bmatrix} | & | & & | \\ W_1 & W_2 & \cdots & W_{d_{out}} \\ | & | & & | \end{bmatrix} = \overset{Z}{\begin{bmatrix} o & o \\ o & \\ & \end{bmatrix}}$$

$$Z_{ij} = \sum_k X_{ik} W_{kj} + b_j = X_i^T W_j + b_j$$

W_j = col of W $\quad X_i^T W + b^T$

$$Z = \begin{bmatrix} X_1^T W_1 + b_1 & X_1^T W_2 + b_2 & \cdots \\ X_2^T W_1 + b_1 & X_2^T W_2 + b_2 & \cdots \\ & & \end{bmatrix}$$

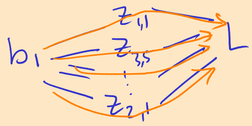
$$Z_{ij} = X_i^T W_j$$

$$Z = XW$$

$$\frac{\partial L}{\partial b_i} = \sum_{j,k} \frac{\partial L}{\partial Z_{jk}} \cdot \frac{\partial Z_{jk}}{\partial b_i} = \begin{cases} 0 & k \neq i \\ 1 & k = i \end{cases}$$

$$= \boxed{\sum_j \frac{\partial L}{\partial Z_{ji}}}$$

$i = k$



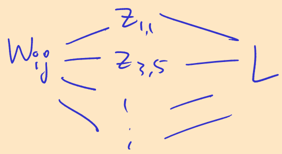
$$\frac{\partial L}{\partial W_{ij}} = \sum_{k,l} \frac{\partial L}{\partial Z_{kl}} \circ \frac{\partial Z_{kl}}{\partial W_{ij}} = \begin{cases} 0 & l \neq j \\ X_{ki} & l = j \end{cases}$$

$$= \sum_k \frac{\partial L}{\partial Z_{kj}} \circ X_{ki}$$

$$= X_{\cdot,i}^T \frac{\partial L}{\partial Z_{\cdot j}}$$

$$Z_{11} = X_{11} W_{11} + X_{12} W_{21} + X_{13} W_{31}$$

$$\frac{\partial Z_{11}}{\partial W_{31}} = X_{13} \quad W_{mn} \quad Z_{kl}$$



$u \in \mathbb{R}^m$ $\quad v \in \mathbb{R}^n$

$A \in \mathbb{R}^{m \times n}$ $\quad A_{ij} = u_i v_j$

$A = uv^T$

$u, v \in \mathbb{R}^n$

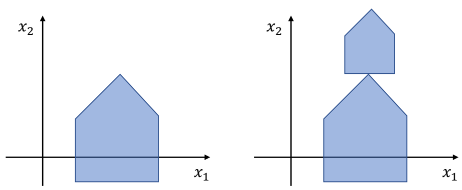
$\sum u_i v_i = u^T v$

CS 189 Introduction to Machine Learning
Spring 2021 Jonathan Shewchuk

DIS11

1 Decision Space

Let's further the intuition about how we can compose arbitrarily complex decision boundaries with a neural network. Consider the images below. For each one, build a network of units with a single output that fires if the input is in the shaded area.



Take-away: MLPs can capture any classification boundary. MLPs are universal classifiers. Note that we haven't said anything yet about their ability to generalize.

2 Backrop in Practice: Staged Computation

For the function $f(x, y, z) = (x + y)z$:

(a) Decompose f into two simpler functions.

(b) Draw the network that represents the computation of f.

(c) Write the forward pass and backward pass (backpropagation) in the network.

(d) Update your network drawing with the intermediate values in the forward and backward pass. Use the inputs $x = -2$, $y = 5$, and $z = -4$.

3 Backpropagation Practice

(a) Chain rule of multiple variables: Assume that you have a function given by $f(x_1, x_2, \ldots, x_n)$, and that $g_i(w) = x_i$ for a scalar variable w. How would you compute $\frac{\mathrm{d}}{\mathrm{d}w} f(g_1(w), g_2(w), \ldots, g_n(w))$? What is its computation graph?

(b) Let $Z = XW + \mathbf{1}b$, where $Z \in \mathbb{R}^{d_n \times d_{out}}$, $X \in \mathbb{R}^{d_n \times d_{in}}$, $W \in \mathbb{R}^{d_{in} \times d_{out}}$, b is a row vector in $\mathbb{R}^{d_{out}}$, and $\mathbf{1}$ is a column vector in $\mathbb{R}^{d_{in}}$. Given $\frac{\partial L}{\partial Z} \in \mathbb{R}^{d_n \times d_{out}}$, where l is a scalar loss, calculate $\frac{\partial L}{\partial W}$ and $\frac{\partial L}{\partial b}$.

4 Model Intuition

(a) What can go wrong if you just initialize all the weights in a neural network to exactly zero? What about to the same nonzero value?

(b) Adding nodes in the hidden layer gives the neural network more approximation ability, because you are adding more parameters. How many weight parameters are there in a neural network with architecture specified by $d = \left[d^{(0)}, d^{(1)}, ..., d^{(N)}\right]$, a vector giving the number of nodes in each of the N layers? Evaluate your formula for a 2 hidden layer network with 10 nodes in each hidden layer, an input of size 8, and an output of size 3.

(c) Consider the two networks in the image below, where the added layer in going from Network A to Network B has 10 units with linear activation. Give one advantage of Network A over Network B, and one advantage of Network B over Network A.