CS 188
Fall 2021

Regular Discussion 11 Solutions

1 Optimization

We would like to classify some data. We have N samples, where each sample consists of a feature vector $\mathbf{x} = [x_1, \cdots, x_k]^T$ and a label $y \in \{0, 1\}$.

Logistic regression produces predictions as follows:

$$P(Y = 1 \mid X) = h(\mathbf{x}) = s\left(\sum_i w_i x_i\right) = \frac{1}{1 + \exp(-(\sum_i w_i x_i))}$$

$$s(\gamma) = \frac{1}{1 + \exp(-\gamma)}$$

where $s(\gamma)$ is the logistic function, $\exp x = e^x$, and $\mathbf{w} = [w_1, \cdots, w_k]^T$ are the learned weights.

Let's find the weights w_j for logistic regression using stochastic gradient descent. We would like to minimize the following loss function (called the cross-entropy loss) for each sample:

$$L = -[y \ln h(\mathbf{x}) + (1 - y) \ln(1 - h(\mathbf{x}))]$$

(a) Show that $s'(\gamma) = s(\gamma)(1 - s(\gamma))$

$$s(\gamma) = (1 + \exp(-\gamma))^{-1}$$
$$s'(\gamma) = -(1 + \exp(-\gamma))^{-2}(-\exp(-\gamma))$$
$$s'(\gamma) = \frac{1}{1 + \exp(-\gamma)} \cdot \frac{\exp(-\gamma)}{1 + \exp(-\gamma)}$$
$$s'(\gamma) = s(\gamma)(1 - s(\gamma))$$

(b) Find $\frac{dL}{dw_j}$. Use the fact from the previous part.

Use chain rule:

$$\frac{dL}{dw_j} = -\left[\frac{y}{h(\mathbf{x})} s'(\sum_i w_i x_i) x_j - \frac{1 - y}{1 - h(\mathbf{x})} s'(\sum_i w_i x_i) x_j\right]$$

Use fact from previous part:

$$\frac{dL}{dw_j} = -\left[\frac{y}{h(\mathbf{x})} h(\mathbf{x})(1 - h(\mathbf{x})) x_j - \frac{1 - y}{1 - h(\mathbf{x})} h(\mathbf{x})(1 - h(\mathbf{x})) x_j\right]$$

Simplify:

$$\begin{aligned}\frac{dL}{dw_j} &= -\left[y(1 - h(\mathbf{x})) x_j - (1 - y) h(\mathbf{x}) x_j\right] \\ &= -x_j[y - y h(\mathbf{x}) - h(\mathbf{x}) + y h(\mathbf{x})] \\ &= -x_j(y - h(\mathbf{x}))\end{aligned}$$

(c) Now, find a simple expression for $\nabla_{\mathbf{w}} L = [\frac{dL}{dw_1}, \frac{dL}{dw_2}, ..., \frac{dL}{dw_k}]^T$

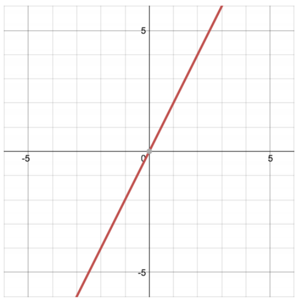
$$\begin{aligned}\nabla_{\mathbf{w}} L &= [-x_1(y - h(\mathbf{x})), -x_2(y - h(\mathbf{x})), ..., -x_k(y - h(\mathbf{x}))]^T \\ &= -[x_1, x_2, ...x_k]^T(y - h(\mathbf{x})) \\ &= -\mathbf{x}(y - h(\mathbf{x}))\end{aligned}$$

(d) Write the stochastic gradient descent update for $\mathbf{w}$. Our step size is η.

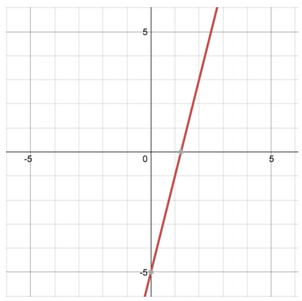
$$\mathbf{w} \leftarrow \mathbf{w} + \eta\mathbf{x}(y - h(\mathbf{x}))$$

2 Neural Network Representations

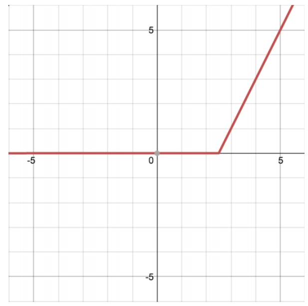
You are given a number of functions (a-h) of a single variable, x, which are graphed below. The computation graphs on the following pages will start off simple and get more complex, building up to neural networks. For each computation graph, indicate which of the functions below they are able to represent.



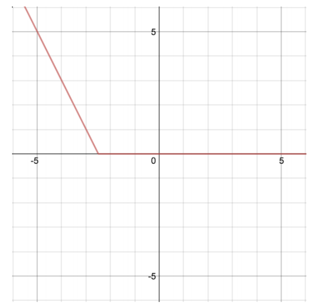
(a) $2x$



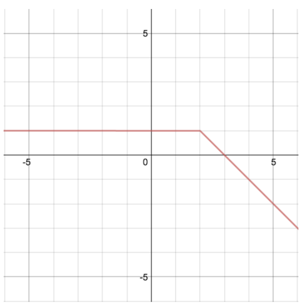
(b) $4x - 5$



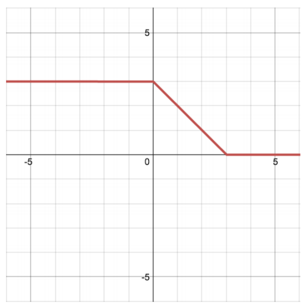
(c) $\begin{cases} 2x - 5 & x \geq 2.5 \\ 0 & x < 2.5 \end{cases}$



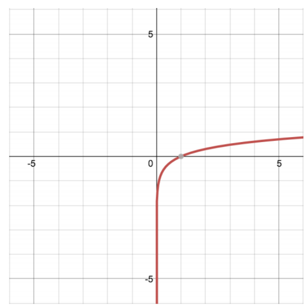
(d) $\begin{cases} -2x - 5 & x \leq -2.5 \\ 0 & x > -2.5 \end{cases}$



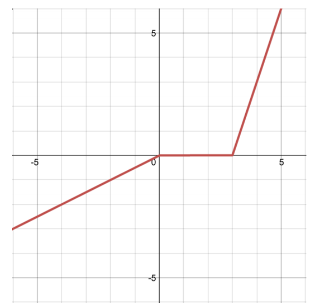
(e) $\begin{cases} -x + 3 & x \geq 2 \\ 1 & x < 2 \end{cases}$



(f) $\begin{cases} 3 & x \leq 0 \\ 3 - x & 0 < x \leq 3 \\ 0 & x > 3 \end{cases}$



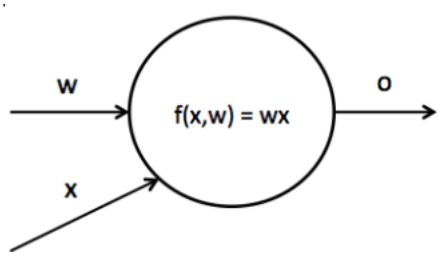
(g) $\log(x)$



(h) $\begin{cases} 0.5x & x \leq 0 \\ 0 & 0 < x \leq 3 \\ 3x - 9 & x > 3 \end{cases}$

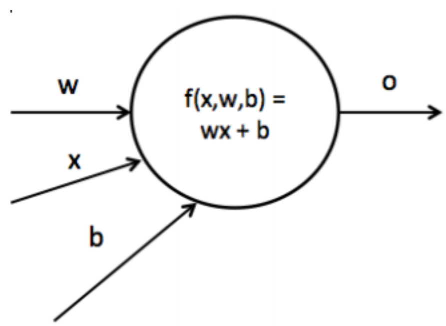
For each of the following computation graphs, determine which functions can be represented by the graph. In parts 1-5, write out the appropriate values of all w's and b's for each function that can be represented.

1. Linear Transformation



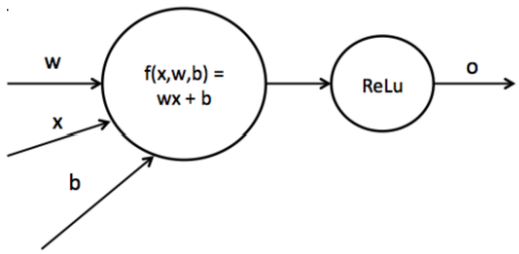
This graph can only represent (a), with $w = 2$. Since there is no bias term, the line must pass through the origin.

2. Linear plus Bias (aka affine transformation)



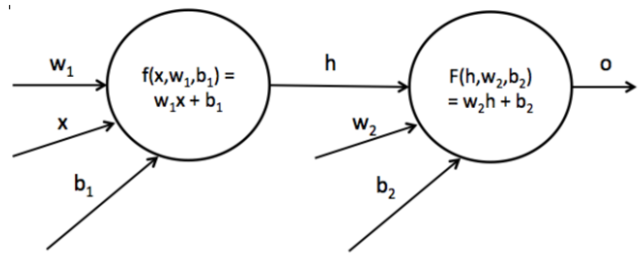
(a) with $w = 2$ and $b = 0$, and (b) with $w = 4$ and $b = -5$

3. Nonlinearity after Linear layer



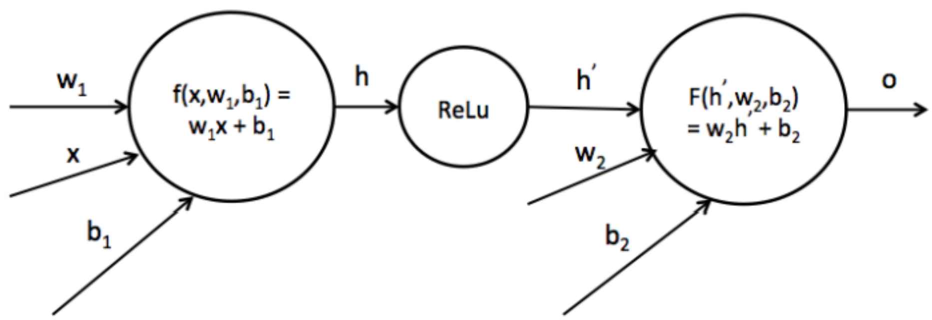
With the output coming directly from the ReLU, this cannot produce any values less than zero. It can produce (c) with $w = 2$ and $b = -5$, and (d) with $w = -2$ and $b = -5$

4. Composition of Affine layers



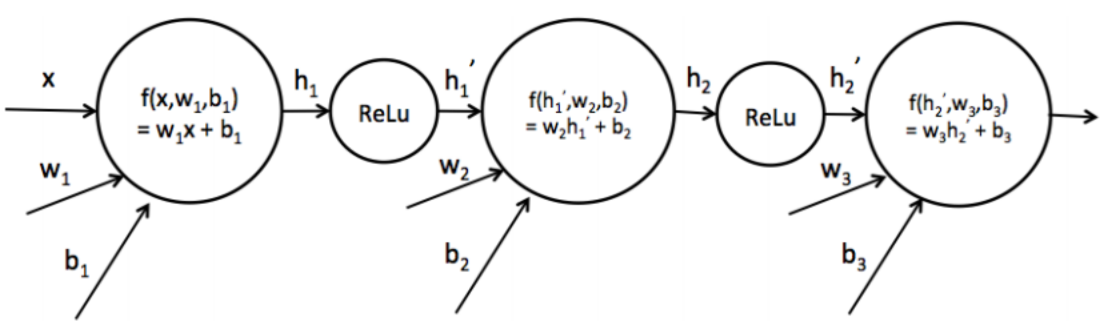
Applying multiple affine transformations (with no non-linearity in between) is not any more powerful than a single affine function: $w_2(w_1x + b_1) + b_2 = w_2w_1x + w_2b_1 + b_2$, so this is just a affine function with different coefficients. The functions we can represent are the same as in 1, if we choose $w_1 = w, w_2 = 1, b_1 = 0, b_2 = b$: (a) with $w_1 = 2, w_2 = 1, b_1 = 0, b_2 = 0$, and (b) with $w_1 = 4, w_2 = 1, b_1 = 0, b_2 = -5$.

5. Two Affine layers with nonlinearity in between (hidden layer)



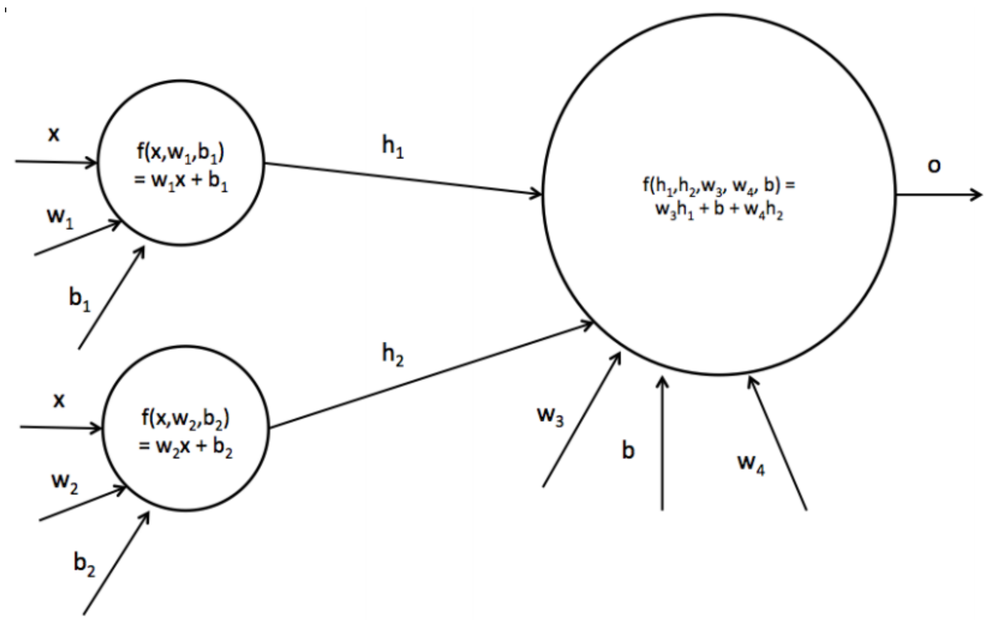
(c), (d), and (e). The affine transformation after the ReLU is capable of stretching (or flipping) and shifting the ReLU output in the vertical dimension. The parameters to produce these are:
(c) with $w_1 = 2, b_1 = -5, w_2 = 1, b_2 = 0$, (d) with $w_1 = -2, b_1 = -5, w_2 = 1, b_2 = 0$, and (e) with $w_1 = 1, b_1 = -2, w_2 = -1, b_2 = 1$

6. Add another hidden layer



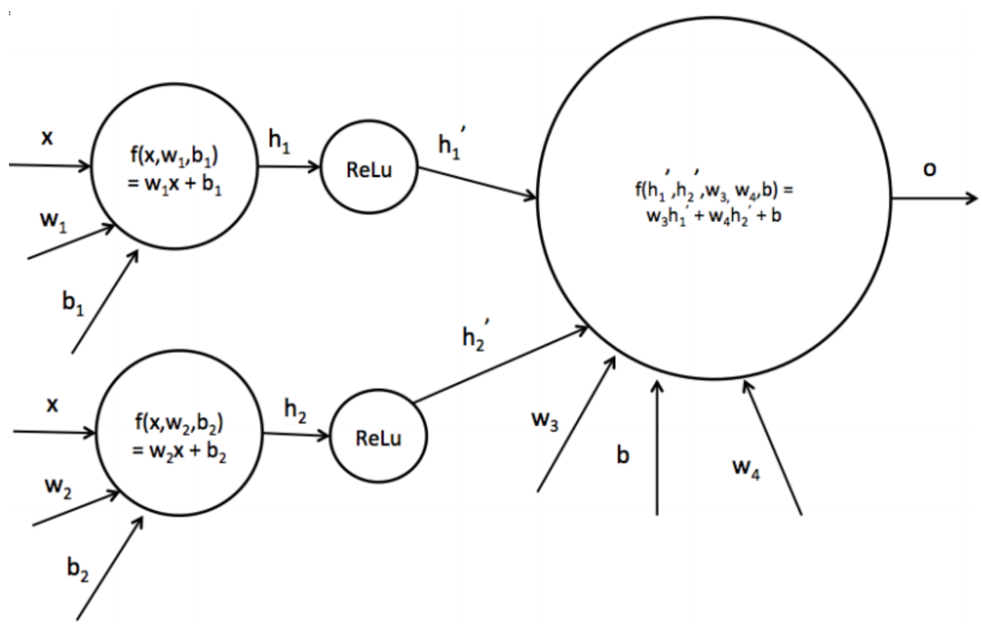
(c), (d), (e), and (f). The network can represent all the same functions as Q5 (because note that we could have $w_2 = 1$ and $b_2 = 0$). In addition it can represent (f): the first ReLU can produce the first flat segment, the affine transformation can flip and shift the resulting curve, and then the second ReLU can produce the second flat segment (with the final affine layer not doing anything). Note that (h) cannot be produced since its line has only one flat segment (and the affine layers can only scale, shift, and flip the graph in the vertical dimension; they can't rotate the graph).

7. Hidden layer of size 2, no nonlinearities



(a) and (b). With no non-linearity, this reduces to a single affine function (in the same way as Q4)

8. Add nonlinearities between layers



All functions except for (g). Note that we can recreate any network from (5) by setting w_4 to 0, so this allows us to produce (c), (d) and (e). To produce the rest of the functions, note that h_1' and h_2' will be two independent functions with a flat part lying on the x-axis, and a portion with positive slope. The final layer takes a weighted sum of these two functions. To produce (a) and (b), the flat portion of one ReLU should start at the point where the other ends ($x = 0$ for (a), or $x = 1$ for (b). The final layer then vertically flips the ReLU sloping down and adds it to the one sloping up, producing a single sloped line. To produce (h), the ReLU sloping down should have its flat portion end (at $x = 0$ before the other's flat portion begins (at $x = 3$). The down-sloping one is again flipped and added to the up-sloping. To produce (f), both ReLUs should have equal slope, which will cancel to produce the first flat portion above the x-axis.