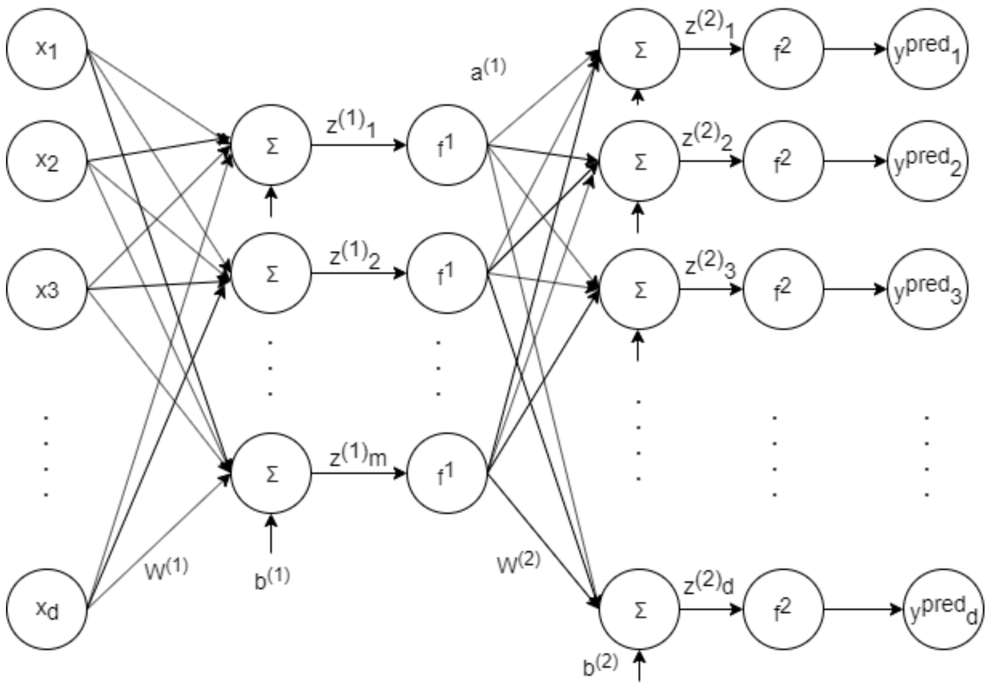
Autoencoder

4. (14 points) Otto N. Coder is exploring different autoencoder architectures. Consider the following autoencoder with input $x \in \mathbb{R}^d$ and output $y^{pred} \in \mathbb{R}^d$. The autoencoder has one hidden layer with m hidden units: $z^{(1)}$, $a^{(1)} \in \mathbb{R}^m$.



$$
\begin{aligned}
z^{(1)} &= W^{(1)}x + b^{(1)} \\
a^{(1)} &= f^{(1)}(z^{(1)}) \quad \text{element-wise} \\
z^{(2)} &= W^{(2)}a^{(1)} + b^{(2)} \\
y^{pred} &= f^{(2)}(z^{(2)}) \quad \text{element-wise}
\end{aligned}
$$

(a) Assume x, $z^{(2)}$, and y^{pred} have dimensions $d \times 1$. Also let $z^{(1)}$ and $a^{(1)}$ have dimensions $m \times 1$. What are the dimensions of the following matrices?

$W^{(1)}$	$b^{(1)}$	$W^{(2)}$	$b^{(2)}$

Otto trains the autoencoder with back-propagation. The loss for a given datapoint x, y is:

$$J(x,y) = \frac{1}{2}||y^{pred} - y||^2 = \frac{1}{2}(y^{pred} - y)^T(y^{pred} - y) \ .$$

Compute the following intermediate partial derivatives. For the following questions, write your answer in terms of x, y, y^{pred}, $W^{(1)}$, $b^{(1)}$, $W^{(2)}$, $b^{(2)}$, $f^{(1)}$, $f^{(2)}$ and **any previously computed or provided partial derivative**. Also note that:

1. Let $\partial f^{(1)}/\partial z^{(1)}$ be an $m \times 1$ matrix, provided to you.
2. Let $\partial f^{(2)}/\partial z^{(2)}$ be a $d \times 1$ matrix, provided to you.
3. If $Ax = y$ where A is a $m \times n$ matrix and x is $n \times 1$ and y is $m \times 1$, then let $\partial y/\partial A = x$.
4. In your answers below, we will assume multiplications are matrix multiplication; to indicate element-wise multiplication, use the symbol $*$.

(b) Find $\partial J/\partial y^{pred}$, a $d \times 1$ matrix.

(c) Find $\partial J/\partial z^{(2)}$, a $d \times 1$ matrix. You may use $\partial J/\partial y^{pred}$ and $*$ for element-wise multiplication.

(d) Find $\partial J/\partial W^{(2)}$, a $d \times m$ matrix. You may use $\partial J/\partial z^{(2)}$.

(e) Write the gradient descent update step for just $W^{(2)}$ for one datapoint (x, y) given learning rate η and $\partial J/\partial W^{(2)}$.

(f) Otto's friend Bigsby believes that bigger is better. He takes a look at Otto's neural network and tells Otto that he should make the number of hidden units m in the hidden layer very large: $m = 10d$. (Recall that $z^{(1)}$ has dimensions $m \times 1$.) Is Bigsby correct? What would you expect to see with training and test accuracy using Bigsby's approach?

(g) Otto's other friend Leila says having more layers is better. Let m be much smaller than d. Leila adds 10 more hidden layers all with linear activation before Otto's current hidden layer (which has sigmoid activation function $f^{(1)}$) such that each hidden layer has m units. What would you expect to see with your training and test accuracy, compared to just having one hidden layer with activation $f^{(1)}$?

(h) Another friend Neil suggests to have several layers with non-linear activation function. He says Otto should regularize the number of active hidden units. Loosely speaking, we consider the average activation of a hidden unit j in our hidden layer 1 (which has sigmoid activation function $f^{(1)}$) to be the average of the activation of $a_j^{(1)}$ over the points x_i in our training dataset of size N:

$$\hat{p}_j = \frac{1}{N} \Sigma_{i=1}^{N} \, a_j^{(1)}(x_i) \; .$$

Assume we would like to enforce the constraint that the average activation for each hidden unit $\hat{p}_j$ is close to some hyperparameter p. Usually, p is very small (say $p < 0.05$).

What is the best format for a regularization penalty given hyperparameter p and the average activation for all our hidden units: $\hat{p}_j$? Select one of the following:

- ◯ Hinge loss: $\Sigma_j \max\left(0, (1 - \hat{p}_j)p\right)$
- ◯ NLL: $\Sigma_j \left(-p \log \frac{p}{\hat{p}_j} - (1-p) \log \frac{(1-p)}{(1-\hat{p}_j)} \right)$
- ◯ Squared loss: $\Sigma_j (\hat{p}_j - p)^2$
- ◯ $l2$ norm: $\Sigma_j (\hat{p}_j)^2$

(i) Which pass should Otto compute $\hat{p}_j$ on? Select one of the following:

- ◯ Forwards pass
- ◯ Backwards pass
- ◯ Gradient descent step (weight update) pass