Tashfeen, Ahmad
CSCI 6213: Data Science Fundamentals

Homework 7

1. Feed Forward Architecture

We denote a collection of p features as a p-dimensional column vector of real numbers.

$$x \in \mathbb{R}^{p \times 1}$$

A simple *feed forward neural network* accepts x as the activations in the input layer $a^{(0)} = x$, feeds these activations past the hidden layers $a^{(1)}, a^{(2)}, a^{(3)}, \dots$ towards the output layer $a^{(L)}$. At last a special function, e. g., $\operatorname{argmax}(a^{(L)})$ is applied to the output layer to get the output of the network. The activations in the subsequent layers past the input layer are calculated via the following feed-forward rule,

$$\begin{aligned} a^{(\ell)} &= \sigma(z^{(\ell)}) \\ z^{(\ell)} &= W^{(\ell)} a^{(\ell-1)} + b^{(\ell)} \end{aligned}$$

The function $\sigma : \mathbb{R} \to (0, 1)$, in this case the sigmoidal, is called the activation function. For all $t \in \mathbb{R}$,

$$\sigma(t) = \frac{1}{1 + e^{-t}}, \quad \frac{\mathrm{d}\,\sigma(t)}{\mathrm{d}\,t} = \sigma(t)(1 - \sigma(t))$$

whereas $W^{(\ell)}$ is a real valued *weight* matrix and $b^{(\ell)}$ is another real valued *bias* vector. Note that $\max(\ell) + 1$ is the total number of layers in the neural network. Training a neural network refers to finding the desired $(W^{(\ell)}, b^{(\ell)})$ for any given problem.

2. Learning Weights and Biases

The natural question to be concerned about at this point is how does one go about figuring out the correct weights and biases $(W^{(\ell)}, b^{(\ell)})$ for each level ℓ? We start by defining the notion of correct, expected or "true" outputs. Let $L = \max(\ell)$ then we can calculate the error in the last layer using the good ol' quadratic error function (sum of the squared differences).

$$\mathcal{E} = (a^{(L)} - t)^2$$

The goal is the same as always, *chop wood; carry water; minimise error.* We begin with random or zero-initialized values for $(W^{(\ell)}, b^{(\ell)})$ and use these to perform a forward pass with an input x from the data to compute $a^{(L)}$. Next, we compare $a^{(L)}$ to the true values t and adjust $(W^{(L)}, b^{(L)})$ accordingly. We then make similar adjustments to $(W^{(L-1)}, b^{(L-1)})$ to achieve the desired $a^{(L-1)}$ and continue this process. This chain of adjustments proceeds recursively backward until we reach and adjust $(W^{(1)}, b^{(1)})$.

An *epoch* is defined as a forward and backwards pass adjustments done for each x in the dataset. We hope that after a certain number of epochs, $(W^{(\ell)}, b^{(\ell)})$ will converge to a local optimum.

3. Backpropagation

The idea that we just defined above with the forward and backward pass is known as the *backpropagation.* The adjustments to $(W^{(\ell)}, b^{(\ell)})$ we spoke of are made via the following learning rule,

$$b^{(\ell)}_{\text{new}} = b^{(\ell)}_{\text{old}} - \eta \left(\frac{\partial \mathcal{E}}{\partial b^{(\ell)}_{\text{old}}} \right), \qquad W^{(\ell)}_{\text{new}} = W^{(\ell)}_{\text{old}} - \eta \left(\frac{\partial \mathcal{E}}{\partial W^{(\ell)}_{\text{old}}} \right)$$

$$\frac{\partial \mathcal{E}}{\partial b^{(\ell)}} = W^{(\ell+1)^\dagger} \left(\frac{\partial \mathcal{E}}{\partial b^{(\ell+1)}} \right) \circ \sigma'(z^{(\ell)}), \qquad \frac{\partial \mathcal{E}}{\partial W^{(\ell)}} = \left(\frac{\partial \mathcal{E}}{\partial b^{(\ell)}} \right) a^{(\ell-1)^\dagger}$$

The hyper-parameter η is known as the learning rate and usually is a small number between 0 and 1.

Bellow we derive the partial derivatives by repeatedly applying the chain rule.

$$
\begin{aligned}
\frac{\partial \mathcal{E}}{\partial b^{(\ell)}} &= \frac{\partial (a^{(L)}-t)^2}{\partial a^{(L)}-t} \circ \frac{\partial \sigma(z^{(L)})}{\partial z^{(L)}} \circ \left(\frac{\partial W^{(L)} a^{(L-1)}}{\partial b^{(\ell)}} + \frac{\partial b^{(L)}}{\partial b^{(\ell)}} \right) \\
&= W^{(L)\dagger} \left(\frac{\partial (a^{(L)}-t)^2}{\partial a^{(L)}-t} \circ \frac{\partial \sigma(z^{(L)})}{\partial z^{(L)}} \right) \circ \left(\frac{\partial a^{(L-1)}}{\partial b^{(\ell)}} \right) \\
&= W^{(L)\dagger} \left(\frac{\partial (a^{(L)}-t)^2}{\partial a^{(L)}-t} \circ \frac{\partial \sigma(z^{(L)})}{\partial z^{(L)}} \right) \circ \left(\frac{\partial \sigma(z^{(L-1)})}{\partial b^{(\ell)}} \right) \\
&= W^{(L)\dagger} \left(\frac{\partial (a^{(L)}-t)^2}{\partial a^{(L)}-t} \circ \frac{\partial \sigma(z^{(L)})}{\partial z^{(L)}} \right) \circ \sigma'(z^{(L-1)}) \circ \left(\frac{\partial z^{(L-1)}}{\partial b^{(\ell)}} \right) \\
&= W^{(L-1)\dagger} \left(W^{(L)\dagger} \left(\frac{\partial (a^{(L)}-t)^2}{\partial a^{(L)}-t} \circ \frac{\partial \sigma(z^{(L)})}{\partial z^{(L)}} \right) \circ \sigma'(z^{(L-1)}) \right) \circ \left(\frac{\partial a^{(L-2)}}{\partial b^{(\ell)}} \right) \\
&= W^{(L-2)\dagger} \left(W^{(L-1)\dagger} \left(W^{(L)\dagger} \left(\frac{\partial (a^{(L)}-t)^2}{\partial a^{(L)}-t} \circ \frac{\partial \sigma(z^{(L)})}{\partial z^{(L)}} \right) \circ \sigma'(z^{(L-1)}) \right) \circ \sigma'(z^{(L-2)}) \right) \circ \left(\frac{\partial a^{(L-3)}}{\partial b^{(\ell)}} \right)
\end{aligned}
$$

Eventually we get,

$$
\begin{aligned}
\frac{\partial \mathcal{E}}{\partial b^{(\ell)}} &= \left(W^{(\ell+1)\dagger} \dots \left(W^{(L-2)\dagger} \left(W^{(L-1)\dagger} \left(W^{(L)\dagger} \left(2(a^{(L)}-t) \circ \sigma'(z^{(L)}) \right) \circ \sigma'(z^{(L-1)}) \right) \circ \sigma'(z^{(L-2)}) \right) \circ \dots \right) \right) \circ \sigma'(z^{(\ell)}) \\
\frac{\partial \mathcal{E}}{\partial W^{(\ell)}} &= \left(W^{(\ell+1)\dagger} \dots \left(W^{(L-2)\dagger} \left(W^{(L-1)\dagger} \left(W^{(L)\dagger} \left(2(a^{(L)}-t) \circ \sigma'(z^{(L)}) \right) \circ \sigma'(z^{(L-1)}) \right) \circ \sigma'(z^{(L-2)}) \right) \circ \dots \right) \circ \sigma'(z^{(\ell)}) \right) a^{(\ell-1)\dagger}
\end{aligned}
$$

4. Three Layer Network

We demonstrate an example three layer feed forward neural network.

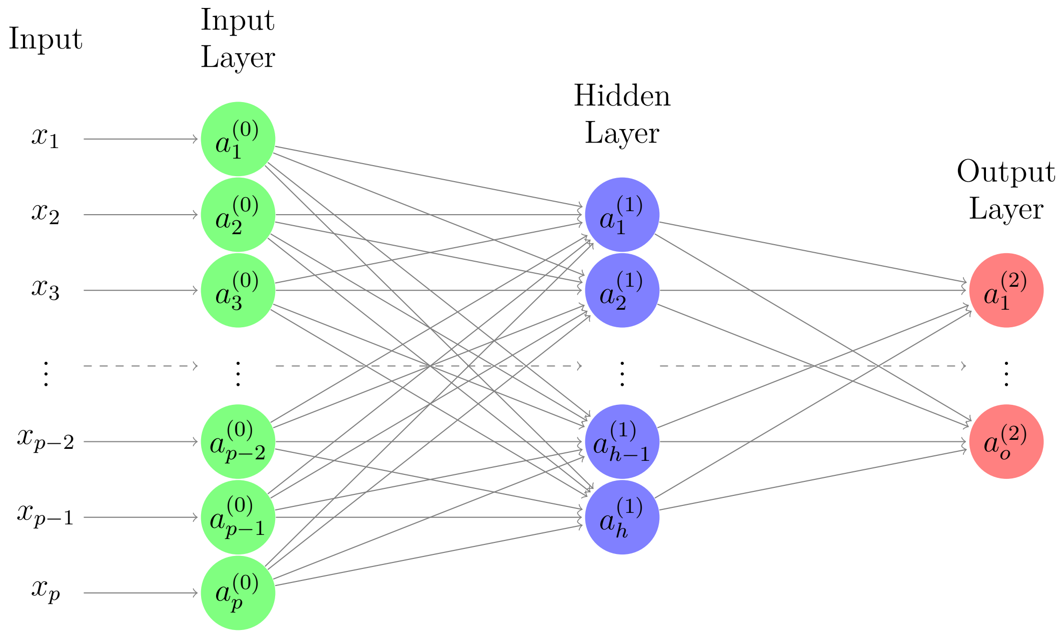

Figure 1. A three layer feed forward neural network.

The network in figure 1 has one input, one hidden and finally one output layer. The activations therein are calculated as follows,

$$
\begin{aligned}
a^{(0)} &= x \\
z^{(1)} &= W^{(1)} a^{(0)} + b^{(1)}, & a^{(1)} &= \sigma(z^{(1)}) \\
z^{(2)} &= W^{(2)} a^{(1)} + b^{(2)}, & a^{(2)} &= \sigma(z^{(2)})
\end{aligned}
$$

The activations $a^{(1)} \in \mathbb{R}^{h\times 1}$ and $a^{(2)} \in \mathbb{R}^{o\times 1}$, in other words for the p dimensional input x, there are h units in the hidden layer and o units in the output layer.

$$W^{(1)} \in \mathbb{R}^{h\times p} \quad \text{and} \quad W^{(2)} \in \mathbb{R}^{o\times h}$$

We return to the learning rules for a three layered network.

$$\begin{aligned} b^{(2)} &= b^{(2)} - \eta\left(\frac{\partial \mathcal{E}}{\partial b^{(2)}}\right) & W^{(2)} &= W^{(2)} - \eta\left(\frac{\partial \mathcal{E}}{\partial W^{(2)}}\right) \\ b^{(1)} &= b^{(1)} - \eta\left(\frac{\partial \mathcal{E}}{\partial b^{(1)}}\right) & W^{(1)} &= W^{(1)} - \eta\left(\frac{\partial \mathcal{E}}{\partial W^{(1)}}\right) \end{aligned}$$

And–the partial derivatives,

$$\begin{aligned} \frac{\partial \mathcal{E}}{\partial b^{(2)}} &= 2(a^{(2)} - t) \circ \sigma'(z^{(2)}) \\ \frac{\partial \mathcal{E}}{\partial W^{(2)}} &= \left(2(a^{(2)} - t) \circ \sigma'(z^{(2)})\right) {a^{(1)}}^{\dagger} \\ \frac{\partial \mathcal{E}}{\partial b^{(1)}} &= \left({W^{(2)}}^{\dagger}\left(2(a^{(2)} - t) \circ \sigma'(z^{(2)})\right)\right) \circ \sigma'(z^{(1)}) \\ \frac{\partial \mathcal{E}}{\partial W^{(1)}} &= \left({W^{(2)}}^{\dagger}\left(2(a^{(2)} - t) \circ \sigma'(z^{(2)})\right) \circ \sigma'(z^{(1)})\right) {a^{(0)}}^{\dagger} \end{aligned}$$

We will now derive these partial derivatives with the chain rule. Recall that,

$$\frac{\mathrm{d}\, f(g(x))}{\mathrm{d}\, x} = \frac{\mathrm{d}\, f(g(x))}{\mathrm{d}\, g(x)} \times \frac{\mathrm{d}\, g(x)}{\mathrm{d}\, x} = \frac{\mathrm{d}\, f}{\mathrm{d}\, g} \times \frac{\mathrm{d}\, g}{\mathrm{d}\, x}$$

Let $u \circ v$ be the *Hadamard product*, we drive the partial derivatives for the output layer weights and biases.

$$\begin{aligned} \frac{\partial \mathcal{E}}{\partial b^{(2)}} &= \frac{\partial (a^{(2)} - t)^2}{\partial b^{(2)}} \\ &= \frac{\partial (a^{(2)} - t)^2}{\partial a^{(2)} - t} \circ \frac{\partial a^{(2)} - t}{\partial b^{(2)}} \\ &= \frac{\partial (a^{(2)} - t)^2}{\partial a^{(2)} - t} \circ \frac{\partial a^{(2)} - t}{\partial b^{(2)}} \\ &= \frac{\partial (a^{(2)} - t)^2}{\partial a^{(2)} - t} \circ \left(\frac{\partial a^{(2)}}{\partial b^{(2)}} - \frac{\partial t}{\partial b^{(2)}}\right) \\ &= \frac{\partial (a^{(2)} - t)^2}{\partial a^{(2)} - t} \circ \frac{\partial a^{(2)}}{\partial b^{(2)}} \\ &= \frac{\partial (a^{(2)} - t)^2}{\partial a^{(2)} - t} \circ \frac{\partial \sigma(z^{(2)})}{\partial b^{(2)}} \\ &= \frac{\partial (a^{(2)} - t)^2}{\partial a^{(2)} - t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \frac{\partial z^{(2)}}{\partial b^{(2)}} \\ &= \frac{\partial (a^{(2)} - t)^2}{\partial a^{(2)} - t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \frac{\partial W^{(2)} a^{(1)} + b^{(2)}}{\partial b^{(2)}} \\ \S 1 &= \frac{\partial (a^{(2)} - t)^2}{\partial a^{(2)} - t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \left(\frac{\partial W^{(2)} a^{(1)}}{\partial b^{(2)}} + \frac{\partial b^{(2)}}{\partial b^{(2)}}\right) \\ &= 2(a^{(2)} - t) \circ \sigma'(z^{(2)})\,(0 + 1) \end{aligned}$$

$$\begin{aligned} \frac{\partial \mathcal{E}}{\partial W^{(2)}} &= \frac{\partial (a^{(2)} - t)^2}{\partial a^{(2)} - t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \left(\frac{\partial W^{(2)} a^{(1)} + b^{(2)}}{\partial W^{(2)}}\right) \\ &= \frac{\partial (a^{(2)} - t)^2}{\partial a^{(2)} - t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \left({a^{(1)}}^{\dagger} + 0\right) \\ &= \left(\frac{\partial (a^{(2)} - t)^2}{\partial a^{(2)} - t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}}\right) {a^{(1)}}^{\dagger} \\ &= \left(2(a^{(2)} - t) \circ \sigma'(z^{(2)})\right) {a^{(1)}}^{\dagger} \end{aligned}$$

By a similar technique we can write down the derivatives for the hidden layer.

$$
\begin{aligned}
\frac{\partial \mathcal{E}}{\partial b^{(1)}} &= \frac{\partial (a^{(2)}-t)^2}{\partial a^{(2)}-t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \circ \left(\frac{\partial W^{(2)} a^{(1)}}{\partial b^{(1)}} + \frac{\partial b^{(2)}}{\partial b^{(1)}} \right) && \text{From } \S1 \\
&= \frac{\partial (a^{(2)}-t)^2}{\partial a^{(2)}-t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \circ \frac{\partial W^{(2)} a^{(1)}}{\partial b^{(1)}} \\
&= W^{(2)\dagger} \left(\frac{\partial (a^{(2)}-t)^2}{\partial a^{(2)}-t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \right) \circ \frac{\partial a^{(1)}}{\partial b^{(1)}} \\
&= W^{(2)\dagger} \left(\frac{\partial (a^{(2)}-t)^2}{\partial a^{(2)}-t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \right) \circ \frac{\partial \sigma(z^{(1)})}{\partial b^{(1)}} \\
&= W^{(2)\dagger} \left(\frac{\partial (a^{(2)}-t)^2}{\partial a^{(2)}-t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \right) \circ \frac{\partial \sigma(z^{(1)})}{\partial z^{(1)}} \frac{\partial z^{(1)}}{\partial b^{(1)}} \\
\S2 &= W^{(2)\dagger} \left(\frac{\partial (a^{(2)}-t)^2}{\partial a^{(2)}-t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \right) \circ \frac{\partial \sigma(z^{(1)})}{\partial z^{(1)}} \frac{\partial W^{(1)} a^{(0)} + b^{(1)}}{\partial b^{(1)}} \\
&= \left(W^{(2)\dagger} \left(2(a^{(2)}-t) \circ \sigma'(z^{(2)}) \right) \right) \circ \sigma'(z^{(1)})(0+1) \\
\frac{\partial \mathcal{E}}{\partial W^{(1)}} &= W^{(2)\dagger} \left(\frac{\partial (a^{(2)}-t)^2}{\partial a^{(2)}-t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \right) \circ \frac{\partial \sigma(z^{(1)})}{\partial z^{(1)}} \frac{\partial W^{(1)} a^{(0)} + b^{(1)}}{\partial W^{(1)}} && \text{From } \S2 \\
&= \left(W^{(2)\dagger} \left(\frac{\partial (a^{(2)}-t)^2}{\partial a^{(2)}-t} \circ \frac{\partial \sigma(z^{(2)})}{\partial z^{(2)}} \right) \circ \frac{\partial \sigma(z^{(1)})}{\partial z^{(1)}} \right) a^{(0)\dagger} \\
&= \left(W^{(2)\dagger} \left(2(a^{(2)}-t) \circ \sigma'(z^{(2)}) \right) \circ \sigma'(z^{(1)}) \right) a^{(0)\dagger}
\end{aligned}
$$

Question 1. $\left(b^{(\ell)}, \partial b^{(\ell)}/\partial \mathcal{E}\right)$ must have the same dimensions. Show the following pairs have the same dimensions,

$$\left(b^{(2)}, \frac{\partial \mathcal{E}}{\partial b^{(2)}}\right), \quad \left(W^{(2)}, \frac{\partial \mathcal{E}}{\partial W^{(2)}}\right), \quad \left(b^{(1)}, \frac{\partial \mathcal{E}}{\partial b^{(1)}}\right), \quad \left(W^{(1)}, \frac{\partial \mathcal{E}}{\partial W^{(1)}}\right)$$

Question 2. Use the chain rule to expand the following derivative,

$$\frac{\mathrm{d}\, f_n(f_{(n-1)}(f_{(n-2)}(\cdots f(x))))}{\mathrm{d}\, x}$$

Question 3. Use the following recursive definition,

$$\frac{\partial \mathcal{E}}{\partial b^{(\ell)}} = W^{(\ell+1)\dagger} \left(\frac{\partial \mathcal{E}}{\partial b^{(\ell+1)}} \right) \circ \sigma'(z^{(\ell)})$$

to obtain the same long-form equation as we did in section 3.

Question 4. Look up "Universal approximation theorem (UAT)" on the internet. Wikipedia is a good starting point. According to the UAT, multilayer feed-forward networks with as few as n hidden layer(s) are universal approximators. What is the value of n?

Question 5. Explore the Anderson's Iris data set. Take 15 Setosas, 15 versicolors and 15 virginicas for your testing set. One-hot encode the training labels and you might want to use `np.expand_dims(X, axis=-1)` to turn the rows of a matrix `X` into column vectors. This is however dependant on your implementation.

1) What is the baseline accuracy of the testing set?
2) Draw a scatter plot of sepal width (cm) against the sepal length (cm). Colour each data point as setosa, versicolor or virginica and label the legends appropriately.
3) Write a neural network from scratch to classify the flowers. How did you pick the hyper parameters?
4) Give the final values of the following,
    (a) Number of hidden layers and number of neural units in each hidden layer.

(b) The learning rate η.
(c) The stopping criteria for training.
(d) The strategy for initialising the weights and biases.
(e) The training error curve.
(f) The final testing accuracy.

You may read in the data as shown in the listing 1.

```
PATH = '../media/data.tsv'
classes = ['setosa', 'versicolor', 'virginica']
data = np.genfromtxt(PATH, delimiter='\t', dtype=str)
features = data[0]
Xy = data[1:].astype(np.float64)
X = Xy[:,:-1]
y = Xy[:,-1].astype(np.int64)
```

Listing 1. Download data.tsv.

Submission Instructions

Submit a `data.py` with all your data processing and a `network.py` with all your neural network, training and testing code. Assure that Numpy and Matplotlib are the only external libraries in use. Finally submit a `<lastname>.pdf` file with all your answers and plots.

Oklahoma City University, Petree College of Arts & Sciences, Computer Science