Question:

Derive systematically the steps involving the backpropagation of a neural network considered in the class with two hidden layers using the cross-entropy loss function. The activation function for each layer can be considered to be **ReLU**.

Notations

- Scalars are denoted by lowercase letters (e.g. x)
- Vectors are denoted by bold lowercase letters (e.g. $\mathbf{x}$), `bold`
- Matrices are denoted by bold uppercase letters (e.g. $\mathbf{X}$), `bold uppercase`

- x_i denotes the i^{th} element of the vector $\mathbf{x}$
- x_{ij} denotes the element in the i^{th} row and j^{th} column of the matrix $\mathbf{X}$
- $\mathbf{x}^{[i]}$ denotes the i^{th} coloumn of the matrix $\mathbf{X}$
- $\mathbf{x}_j$ denotes the j^{th} row of the matrix

- $n^{[l]}$ denotes the number of neurons in the l^{th} layer
- n_x denotes the number of features in the input layer
- n_y denotes the number of nuerons in the output layer which is equal to the number of classes for classification problems
- m denotes the number of training examples

Shapes

- $\mathbf{X} \in \mathbb{R}^{n_x \times m}$, is a input matrix.
- $\hat{\mathbf{Y}} \in \mathbb{R}^{n_y \times m}$, is a predicted output matrix.

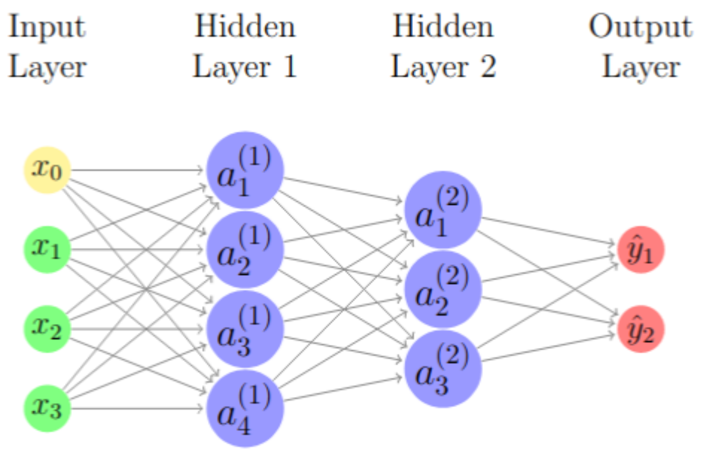
$$\hat{Y} = \mathbf{W}^{[L]}\mathbf{A}^{[L-1]} + \mathbf{b}^{[L]}$$

- $\mathbf{Y} \in \mathbb{R}^{n_y \times m}$, is a output matrix.
- $\mathbf{W}^{[l]} \in \mathbb{R}^{n^{[l]} \times n^{[l-1]}}$, is a weight matrix.
- $\mathbf{b}^{[l]} \in \mathbb{R}^{n^{[l]} \times 1}$, is a bias vector.
- $\mathbf{A}^{[l]} \in \mathbb{R}^{n^{[l]} \times m}$, is a output matrix.
- $\mathbf{Z}^{[l]} \in \mathbb{R}^{n^{[l]} \times m}$, is a matrix of linear activations.

$$\mathbf{Z}^{[l]} = \mathbf{W}^{[l]}\mathbf{A}^{[l-1]} + \mathbf{b}^{[l]}$$

$$\mathbf{A}^{[0]} = \mathbf{X}, \quad \text{which is input}$$

- $g^{[l]}$, is the activation function of the l^{th} layer.
- $\mathbf{A}^{[l]} = g^{[l]}(\mathbf{Z}^{[l]})$, is a matrix of activations where $g^{[l]}$ is the activation function of the l^{th} layer.



The neural network architecture

Initialization

- We have input of shape (n_x, m)
- **First hidden layer** : we have $n^{[1]}$ nuerons. We initialize the weights and bias as follows:

$$\mathbf{W}^{[1]} \in \mathbb{R}^{n^{[1]} \times n_x}$$

$$\mathbf{b}^{[1]} \in \mathbb{R}^{n^{[1]} \times 1}$$

- **Second hidden layer** : we have $n^{[2]} neurons$. We initialize the weights and bias as follows:

$$\mathbf{W}^{[2]} \in \mathbb{R}^{n^{[2]} \times n^{[1]}}$$

$$\mathbf{b}^{[2]} \in \mathbb{R}^{n^{[2]} \times 1}$$

- **Output layer** : we have $n^{[3]}$ or n_y nuerons. We initialize the weights and bias as follows:

$$\mathbf{W}^{[3]} \in \mathbb{R}^{n^{[3]} \times n^{[2]}}$$

$$\mathbf{b}^{[3]} \in \mathbb{R}^{n^{[3]} \times 1}$$

Forward Propagation

1. At the `first layer`, we have $A^{[0]} = \mathbf{X}$, which is the input.

- we will calculate the linear activation $\mathbf{Z}^{[1]}$ and the activation $\mathbf{A}^{[1]}$ as follows:

$$\mathbf{Z}^{[1]} = \mathbf{W}^{[1]}\mathbf{A}^{[0]} + \mathbf{b}^{[1]}$$
$$\mathbf{A}^{[1]} = g^{[1]}(\mathbf{Z}^{[1]})$$

2. At the `second layer`, we have $\mathbf{A}^{[1]}$ as the input.

- we will calculate the linear activation $\mathbf{Z}^{[2]}$ and the activation $\mathbf{A}^{[2]}$ as follows:

$$\mathbf{Z}^{[2]} = \mathbf{W}^{[2]}A^{[1]} + \mathbf{b}^{[2]}$$
$$\mathbf{A}^{[2]} = g^{[2]}(\mathbf{Z}^{[2]})$$

3. Now at the `output layer`, we have $\mathbf{A}^{[3]}$ as the input.

- we will calculate the linear activation $\mathbf{Z}^{[3]}$ and the activation $\mathbf{A}^{[3]}$ as follows:

$$\mathbf{Z}^{[3]} = \mathbf{W}^{[3]}\mathbf{A}^{[2]} + \mathbf{b}^{[3]}$$
$$\mathbf{A}^{[3]} = g^{[3]}(\mathbf{Z}^{[3]})$$

$g^{[3]}$ is the activation function of the output layer. For classification problems, we use the softmax function. For regression problems, we use the identity function.

This $\mathbf{A}^{[3]}$ is the output of the neural network.

It can be seen as the function of it's previous layers parameters and the input $\mathbf{X}$.

$$\mathbf{A}^{[3]} = f(\mathbf{X}, \mathbf{W}^{[1]}, \mathbf{b}^{[1]}, \mathbf{W}^{[2]}, \mathbf{b}^{[2]}, \mathbf{W}^{[3]}, \mathbf{b}^{[3]})$$
$$\mathbf{Y} = f(g^{[3]}(g^{[2]}(g^{[1]}(\mathbf{X}, \mathbf{W}^{[1]}, \mathbf{b}^{[1]}), \mathbf{W}^{[2]}, \mathbf{b}^{[2]}), \mathbf{W}^{[3]}, \mathbf{b}^{[3]}))$$

Backward Propagation

- Now we have to update the parameters of the neural network to minimize the loss function.
- We will use the `cross-entropy loss` function.

$$J = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(\hat{y}_k^{(i)})$$

this is a scalar value.

here shape of $\mathbf{Y}$ is (n_y, m) and shape of $\hat{\mathbf{Y}}$ is (n_y, m)

so $y_k^{(i)}$ is the k^{th} element of the i^{th} sample of $\mathbf{Y}$

and $\hat{y}_k^{(i)}$ is the k^{th} element of the i^{th} sample of $\hat{\mathbf{Y}}$

Gradient of the loss function

- We will use the gradient descent algorithm to update the parameters.
- We will use the chain rule to calculate the gradients of the loss function with respect to the parameters.
- As Loss `J` is a function of the parameters `W` and `b`, we can calculate the gradients of `J` with respect to `W` and `b` as follows:

$$\frac{\partial J}{\partial \mathbf{W}^{[l]}} = \frac{\partial J}{\partial \mathbf{A}^{[l]}} \frac{\partial \mathbf{A}^{[l]}}{\partial \mathbf{Z}^{[l]}} \frac{\partial \mathbf{Z}^{[l]}}{\partial \mathbf{W}^{[l]}}$$

$$\frac{\partial J}{\partial \mathbf{b}^{[l]}} = \frac{\partial J}{\partial \mathbf{A}^{[l]}} \frac{\partial \mathbf{A}^{[l]}}{\partial \mathbf{Z}^{[l]}} \frac{\partial \mathbf{Z}^{[l]}}{\partial \mathbf{b}^{[l]}}$$

Output layer

- We will calculate the gradient of the loss function `(categorical-cross entropy)` with respect to the $A^{[3]}$ as follows:

$$\frac{\partial J}{\partial \mathbf{A}^{[3]}} = -\frac{1}{m}(\frac{\mathbf{Y}}{\hat{\mathbf{Y}}})$$

- Now we will calculate the gradients of the $\mathbf{A}^{[3]}$ `(softmax)` with respect to the $\mathbf{Z}^{[3]}$ as follows:

$$\frac{\partial \mathbf{A_i}^{[3]}}{\partial \mathbf{Z_j}^{[3]}} = \mathbf{A_i}^{[3]}(\delta_{ij} - \mathbf{A_i}^{[3]})$$

As we are using the `softmax activation` function at the output layer.

- Now we will calculate the gradients of the $\mathbf{Z}^{[3]}$ with respect to the $\mathbf{W}^{[3]}$ and $\mathbf{b}^{[3]}$ as follows:

$$\frac{\partial \mathbf{Z}^{[3]}}{\partial \mathbf{W}^{[3]}} = \mathbf{A}^{[2]}$$

$$\frac{\partial \mathbf{Z}^{[3]}}{\partial \mathbf{b}^{[3]}} = 1$$

hence we have the following equations:

$$\frac{\partial J}{\partial \mathbf{b}^{[3]}} = \frac{\partial J}{\partial \mathbf{A}^{[3]}} \frac{\partial \mathbf{A}^{[3]}}{\partial \mathbf{Z}^{[3]}} \frac{\partial \mathbf{Z}^{[3]}}{\partial \mathbf{b}^{[3]}}$$

$$= (-\frac{1}{m}(\frac{\mathbf{Y}}{\hat{\mathbf{Y}}})) * (\mathbf{A_i}^{[3]}(\delta_{ij} - \mathbf{A_i}^{[3]})) * (1) = -\frac{1}{m}(\hat{\mathbf{Y}} - \mathbf{Y}) * (1) = \delta^{[3]}$$

$$\frac{\partial J}{\partial \mathbf{W}^{[3]}} = \delta^{[3]} * (\mathbf{A}^{[2]})$$

where $\delta^{[3]}$ is the error of the output layer.

Now to update the weights and bias of the output layer, we will use the following equations:

$$\mathbf{W}^{[3]} = \mathbf{W}^{[3]} - \alpha \frac{\partial J}{\partial W^{[3]}}$$

$$\mathbf{b}^{[3]} = \mathbf{b}^{[3]} - \alpha \frac{\partial J}{\partial b^{[3]}}$$

where α is the learning rate.

Here we will pass the $\delta^{[3]}$ to the previous layer by multiplying it with the weights of this layer.

$$\frac{\partial J}{\partial \mathbf{A}^{[2]}} = \frac{\partial J}{\partial \mathbf{A}^{[3]}} \frac{\partial \mathbf{A}^{[3]}}{\partial \mathbf{Z}^{[3]}} \frac{\partial \mathbf{Z}^{[3]}}{\partial \mathbf{A}^{[2]}}$$

$$= \delta^{[3]} * (\mathbf{W}^{[3]})$$

Second hidden layer

Here we will update the weights and bias of the second hidden layer. i.e. $\mathbf{W}^{[2]}$ and $\mathbf{b}^{[2]}$.

For that we need to calculate the gradients of the loss function with respect to the $\mathbf{W}^{[2]}$ and $\mathbf{b}^{[2]}$.

So, we will calculate the gradients of the $\mathbf{A}^{[2]}$ with respect to the $\mathbf{Z}^{[2]}$ as follows: as activation function at the second hidden layer is relu, so we will use the following equation:

$$\frac{\partial \mathbf{A}^{[2]}}{\partial \mathbf{Z}^{[2]}} = \begin{cases} 1 & \text{if } \mathbf{Z}^{[2]} > 0 \\ 0 & \text{otherwise} \end{cases}$$

Now we will calculate the gradients of the $\mathbf{Z}^{[2]}$ with respect to the $\mathbf{W}^{[2]}$ and $\mathbf{b}^{[2]}$ as follows:

$$\frac{\partial \mathbf{Z}^{[2]}}{\partial \mathbf{W}^{[2]}} = \mathbf{A}^{[1]}$$

$$\frac{\partial \mathbf{Z}^{[2]}}{\partial \mathbf{b}^{[2]}} = 1$$

So we have the following equations:

$$\frac{\partial J}{\partial \mathbf{b}^{[2]}} = \frac{\partial J}{\partial \mathbf{A}^{[3]}} \frac{\partial \mathbf{A}^{[3]}}{\partial \mathbf{Z}^{[3]}} \frac{\partial \mathbf{Z}^{[3]}}{\partial \mathbf{A}^{[2]}} \frac{\partial \mathbf{A}^{[2]}}{\partial \mathbf{Z}^{[2]}} \frac{\partial \mathbf{Z}^{[2]}}{\partial \mathbf{b}^{[2]}}$$

$$= \delta^{[3]} * (\mathbf{W}^{[3]}) * (\begin{cases} 1 & \text{if } \mathbf{Z}^{[2]} > 0 \\ 0 & \text{otherwise} \end{cases}) * (1) = \delta^{[2]}$$

$$\frac{\partial J}{\partial \mathbf{W}^{[2]}} = \delta^{[2]} * (\frac{\partial \mathbf{Z}^{[2]}}{\partial \mathbf{W}^{[2]}})$$

$$= \delta^{[2]} * (\mathbf{A}^{[1]})$$

where $\delta^{[2]}$ is the error of the second hidden layer.

Now to update the weights and bias of the second hidden layer, we will use the following equations:

$$\mathbf{W}^{[2]} = \mathbf{W}^{[2]} - \alpha \frac{\partial J}{\partial W^{[2]}}$$

$$\mathbf{b}^{[2]} = \mathbf{b}^{[2]} - \alpha \frac{\partial J}{\partial b^{[2]}}$$

where α is the learning rate.

Here we will pass the $\delta^{[2]}$ to the previous layer by multiplying it with the weights of this layer.

$$\frac{\partial J}{\partial \mathbf{A}^{[1]}} = \frac{\partial J}{\partial \mathbf{A}^{[3]}} \frac{\partial \mathbf{A}^{[3]}}{\partial \mathbf{Z}^{[3]}} \frac{\partial \mathbf{Z}^{[3]}}{\partial \mathbf{A}^{[2]}} \frac{\partial \mathbf{A}^{[2]}}{\partial \mathbf{Z}^{[2]}} \frac{\partial \mathbf{Z}^{[2]}}{\partial \mathbf{A}^{[1]}}$$

$$= \delta^{[2]} * (\mathbf{W}^{[2]})$$

First hidden layer

Here we will update the weights and bias of the first hidden layer. i.e. $\mathbf{W}^{[1]}$ and $\mathbf{b}^{[1]}$.

For that we need to calculate the gradients of the loss function with respect to the $\mathbf{W}^{[1]}$ and $\mathbf{b}^{[1]}$.

So, we will calculate the gradients of the $\mathbf{A}^{[1]}$ with respect to the $\mathbf{Z}^{[1]}$ as follows:

as activation function at the first hidden layer is relu, so we will use the following equation:

$$\frac{\partial \mathbf{A}^{[1]}}{\partial \mathbf{Z}^{[1]}} = \begin{cases} 1 & \text{if } \mathbf{Z}^{[1]} > 0 \\ 0 & \text{otherwise} \end{cases}$$

Now we will calculate the gradients of the $\mathbf{Z}^{[1]}$ with respect to the $\mathbf{W}^{[1]}$ and $\mathbf{b}^{[1]}$ as follows:

$$\frac{\partial \mathbf{Z}^{[1]}}{\partial \mathbf{W}^{[1]}} = \mathbf{A}^{[0]} = \mathbf{X}$$

$$\frac{\partial \mathbf{Z}^{[1]}}{\partial \mathbf{b}^{[1]}} = 1$$

So we have the following equations:

$$\frac{\partial J}{\partial \mathbf{b}^{[1]}} = \frac{\partial J}{\partial \mathbf{A}^{[3]}} \frac{\partial \mathbf{A}^{[3]}}{\partial \mathbf{Z}^{[3]}} \frac{\partial \mathbf{Z}^{[3]}}{\partial \mathbf{A}^{[2]}} \frac{\partial \mathbf{A}^{[2]}}{\partial \mathbf{Z}^{[2]}} \frac{\partial \mathbf{Z}^{[2]}}{\partial \mathbf{A}^{[1]}} \frac{\partial \mathbf{A}^{[1]}}{\partial \mathbf{Z}^{[1]}} \frac{\partial \mathbf{Z}^{[1]}}{\partial \mathbf{b}^{[1]}}$$

$$= \delta^{[2]} * (\mathbf{W}^{[2]}) * (\begin{cases} 1 & \text{if } \mathbf{Z}^{[1]} > 0 \\ 0 & \text{otherwise} \end{cases}) * (1) = \delta^{[1]}$$

$$\frac{\partial J}{\partial \mathbf{W}^{[1]}} = \delta^{[1]} * (\frac{\partial \mathbf{Z}^{[1]}}{\partial \mathbf{W}^{[1]}})$$

$$= \delta^{[1]} * (\mathbf{A}^{[0]}) = \delta^{[1]} * (\mathbf{X})$$

where $\delta^{[1]}$ is the error of the first hidden layer.

Now to update the weights and bias of the first hidden layer, we will use the following equations:

$$\mathbf{W}^{[1]} = \mathbf{W}^{[1]} - \alpha \frac{\partial J}{\partial W^{[1]}}$$

$$\mathbf{b}^{[1]} = \mathbf{b}^{[1]} - \alpha \frac{\partial J}{\partial b^{[1]}}$$

where α is the learning rate.

So we have now updated the weights and bias of all the layers.

Now we will repeat the above steps for all the training examples and then update the weights and bias of all the layers.

Generalization

1. Forward propagation

$$\mathbf{A}^{[0]} = \mathbf{X}$$
$$\mathbf{Z}^{[l]} = \mathbf{W}^{[l]}\mathbf{A}^{[l-1]} + \mathbf{b}^{[l]}$$
$$\mathbf{A}^{[l]} = g^{[l]}(\mathbf{Z}^{[l]})$$

where $g^{[l]}$ is the activation function of the l^{th} layer. and l=1,2,3,4...

2. Backward propagation

- We have Loss function as Cross entropy loss function.

$$J = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(\hat{y}_k^{(i)})$$

So gradient of Loss function with respect to $\mathbf{A}^{[L]}$ is:

$$\frac{\partial J}{\partial \mathbf{A}^{[L]}} = -\frac{1}{m}\sum_{i=1}^{m}\frac{y^{(i)}}{\hat{y}^{(i)}}$$

- General updating rule for the weights and bias of the l^{th} layer is:

$$\mathbf{W}^{[l]} = \mathbf{W}^{[l]} - \alpha\frac{\partial J}{\partial W^{[l]}}$$
$$\mathbf{b}^{[l]} = \mathbf{b}^{[l]} - \alpha\frac{\partial J}{\partial b^{[l]}}$$

here $\frac{\partial J}{\partial W^{[l]}}$ is the gradient of the loss function with respect to the weights of the l^{th} layer.

$$\frac{\partial J}{\partial W^{[l]}} = \frac{\partial J}{\partial \mathbf{A}^{[L]}}\frac{\partial \mathbf{A}^{[L]}}{\partial \mathbf{Z}^{[L]}}\frac{\partial \mathbf{Z}^{[L]}}{\partial \mathbf{A}^{[L-1]}}\cdots\frac{\partial \mathbf{Z}^{[l]}}{\partial \mathbf{W}^{[l]}}$$
$$\frac{\partial J}{\partial b^{[l]}} = \frac{\partial J}{\partial \mathbf{A}^{[L]}}\frac{\partial \mathbf{A}^{[L]}}{\partial \mathbf{Z}^{[L]}}\frac{\partial \mathbf{Z}^{[L]}}{\partial \mathbf{A}^{[L-1]}}\cdots\frac{\partial \mathbf{Z}^{[l]}}{\partial \mathbf{b}^{[l]}}$$
$$\frac{\partial J}{\partial \mathbf{A}^{[L]}} = -\frac{1}{m}\sum_{i=1}^{m}\frac{y^{(i)}}{\hat{y}^{(i)}}$$
$$\frac{\partial \mathbf{A}^{[l]}}{\partial \mathbf{Z}^{[l]}} = g^{[l]'}(\mathbf{Z}^{[l]})\ and\ \frac{\partial \mathbf{Z}^{[l]}}{\partial \mathbf{A}^{[l-1]}} = \mathbf{W}^{[l]}$$

where $\frac{\partial \mathbf{Z}^{[l]}}{\partial \mathbf{W}^{[l]}} = \mathbf{A}^{[l-1]}$ and $\frac{\partial \mathbf{Z}^{[l]}}{\partial \mathbf{b}^{[l]}} = 1$

So we can find the $\frac{\partial J}{\partial \mathbf{W}^{[l]}}$ and $\frac{\partial J}{\partial \mathbf{b}^{[l]}}$ for any layer l and then update the weights and bias in the backpropagation step. and then repeat this again and again.