Ryan Ott
14862565
ryan.ott@student.uva.nl



# 1 Linear Module and Activation Module

## 1.a Linear Module

$$\left[\frac{\partial L}{\partial \mathbf{W}}\right]_{ij} = \frac{\partial L}{\partial W_{ij}} = \sum_{s,n} \frac{\partial L}{\partial Y_{sn}} \frac{\partial Y_{sn}}{\partial W_{ij}} \tag{1}$$

$$\frac{\partial Y_{sn}}{\partial W_{ij}} = \frac{\partial}{\partial W_{ij}} \left( \sum_m [\mathbf{X}]_{sm} [\mathbf{W}^\top]_{mn} + [\mathbf{B}]_{sn} \right) = \sum_m X_{sm} \frac{\partial W_{nm}}{\partial W_{ij}} + \frac{\partial B_{sn}}{\partial W_{ij}} \tag{2}$$

$$= \sum_m X_{sm} \delta_{ni} \delta_{mj} + 0 = X_{sj} \delta_{ni} \tag{3}$$

$$\sum_{s,n} \frac{\partial L}{\partial Y_{sn}} \frac{\partial Y_{sn}}{\partial W_{ij}} = \sum_{s,n} \frac{\partial L}{\partial Y_{sn}} X_{sj} \delta_{ni} = \sum_s \frac{\partial L}{\partial Y_{si}} X_{sj} \tag{4}$$

$$\therefore \frac{\partial L}{\partial \mathbf{W}} = \left( \frac{\partial L}{\partial \mathbf{Y}} \right)^\top \mathbf{X} \quad \in \mathbb{R}^{N \times M} \tag{5}$$

## 1.b

$$\left[\frac{\partial L}{\partial \mathbf{b}}\right]_{j} = \frac{\partial L}{\partial b_j} = \sum_{s,n} \frac{\partial L}{\partial Y_{sn}} \frac{\partial Y_{sn}}{\partial b_j} \tag{6}$$

$$\frac{\partial Y_{sn}}{\partial b_j} = \frac{\partial}{\partial b_j} \left( \sum_m [\mathbf{X}]_{sm} [\mathbf{W}^\top]_{mn} + [\mathbf{B}]_{sn} \right) = \sum_m \frac{\partial X_{sm} W_{nm}}{\partial b_j} + \frac{\partial B_{sn}}{\partial b_j} \tag{7}$$

$$= 0 + \delta_{nj} = \delta_{nj} \tag{8}$$

$$\sum_{s,n} \frac{\partial L}{\partial Y_{sn}} \frac{\partial Y_{sn}}{\partial b_j} = \sum_{s,n} \frac{\partial L}{\partial Y_{sn}} \delta_{nj} = \sum_s \frac{\partial L}{\partial Y_{sj}} \tag{9}$$

$$\therefore \frac{\partial L}{\partial \mathbf{b}} = \sum_s \frac{\partial L}{\partial \mathbf{Y}_s} \quad \in \mathbb{R}^{1 \times N} \tag{10}$$

For clarification, in equation (10) we sum over the rows $s \in S$ of $\mathbf{Y}$, so the $j$-th element of $\mathbf{b}$ is the sum of all elements in position $j$ of the rows of $\mathbf{Y}$.

## 1.c

$$\left[\frac{\partial L}{\partial \mathbf{X}}\right]_{ij} = \frac{\partial L}{\partial X_{ij}} = \sum_{s,n} \frac{\partial L}{\partial Y_{sn}} \frac{\partial Y_{sn}}{\partial X_{ij}} \tag{11}$$

$$\frac{\partial Y_{sn}}{\partial X_{ij}} = \frac{\partial}{\partial X_{ij}} \left( \sum_m [\mathbf{X}]_{sm} [\mathbf{W}^\top]_{mn} + [\mathbf{B}]_{sn} \right) = \sum_m \frac{\partial X_{sm}}{\partial X_{ij}} W_{nm} + \frac{\partial B_{sn}}{\partial X_{ij}} \tag{12}$$

$$= \sum_m \delta_{si} \delta_{mj} W_{nm} + 0 = \delta_{si} W_{nj} \tag{13}$$

$$\sum_{s,n} \frac{\partial L}{\partial Y_{sn}} \frac{\partial Y_{sn}}{\partial X_{ij}} = \sum_{s,n} \frac{\partial L}{\partial Y_{sn}} \delta_{si} W_{nj} = \sum_n \frac{\partial L}{\partial Y_{in}} W_{nj} \tag{14}$$

$$\therefore \frac{\partial L}{\partial \mathbf{X}} = \frac{\partial L}{\partial \mathbf{Y}} \mathbf{W} \quad \in \mathbb{R}^{S \times M} \tag{15}$$

### 1.d Activation Module

$$\left[\frac{\partial L}{\partial \mathbf{X}}\right]_{ij} = \frac{\partial L}{\partial X_{ij}} = \sum_{s,m} \frac{\partial L}{\partial Y_{sm}} \frac{\partial Y_{sm}}{\partial X_{ij}} \tag{16}$$

$$= \sum_{s,m} \frac{\partial L}{\partial Y_{sm}} \frac{\partial h(X_{sm})}{\partial X_{ij}} \tag{17}$$

$$= \sum_{s,m} \frac{\partial L}{\partial Y_{sm}} \delta_{si}\delta_{mj} = \frac{\partial L}{\partial Y_{ij}} \tag{18}$$

$$\therefore \frac{\partial L}{\partial \mathbf{X}} = \frac{\partial L}{\partial \mathbf{Y}} \circ h'(\mathbf{X}) \quad \in \mathbb{R}^{S \times M} \tag{19}$$

From equation (17) to (18) we get two Kronecker deltas because the derivative of the element-wise activation function is zero for all elements except the one we are taking the derivative of. This makes intuitive sense, given we are differentiating an element-wise function. The final derivative of the loss w.r.t. the input $\mathbf{X}$ is the Hadamard product of the derivative of the loss w.r.t. the output $\mathbf{Y}$ and the element-wise derivative of the activation function $h$ w.r.t. the input $\mathbf{X}$. We can assume the shapes of $\mathbf{X}$ and $\mathbf{Y}$ are compatible, since $\mathbf{Y} = h(\mathbf{X})$

## 2 Softmax, Loss Modules and Residuals

Let

### 2.a Softmax

Ligma

### 2.b Residuals

*Which constraints does the residual connection place on $N_1$ and $N_1$, the numbers of neurons in the two linear layers of the LAL module?*
The first linear layer $L_1$ must have have $F$ rows to allow for the matrix multiplication with the input $\mathbf{X} \in \mathbb{R}^{S \times F}$. It will then have an arbitrary output dimension (that is compatible with the activation function). The second linear layer $L_2$ must however have the exact same shape as $X$ to allow for the element-wise addition of the residual connection. So $N_2 = S \times F$.

### 2.c

*How does adding the residual connection change $\frac{\partial L}{\partial \mathbf{X}}$?*
The derivative of the loss now has an additional term, which is the derivative of the residual connection. This is the identity function, so the derivative is 1. The derivative of the loss w.r.t. the input $\mathbf{X}$ is now the sum of the derivative of the loss w.r.t. the output $\mathbf{Y}$ plus the derivative of the residual connection w.r.t. the input $\mathbf{X}$.

$$\frac{\partial L}{\partial \mathbf{X}} = \frac{\partial L}{\partial \mathbf{Y}}\mathbf{W} + \frac{\partial \mathbf{X}}{\partial \mathbf{X}} \tag{20}$$

$$= \frac{\partial L}{\partial \mathbf{Y}}\mathbf{W} + \mathbf{I} \tag{21}$$

### 2.d

*Briefly explain how your answer to (c) improves the stability of training a deep neural network made up of many such residual blocks, also known as ResNet.*

- **Vanishing gradients:** By adding residual connections, the gradient of the loss w.r.t. the input $\mathbf{X}$ can take an alternative path through the network and is far less likely to vanish. Applying the chain rule many times makes us multiply small numbers by each other, making the gradient smaller as it propagates. By adding the input element-wise, we keep an alternative, non-vanishing path in the backward.

- **Reduced Error Propagation:** The network is able to learn more complex functions and isn't limited to passing on the errornous output of the previous layer. It can learn more low-level features from the input or highly process them to learn high-level features.
- **Smoother Loss Landscape:** Because the gradients dont vanish as easily, there are less sharp edges in the loss surface. Also, by allowing both high and low-level features to be passed on, the problem can be modeled better, thus smoothing the loss surface.

# 3 NumPy Implementation

Test accuracy was 49.88% using the default settings. We see that the training loss is monotonically decreasing, which is what we hope for. The network is learning something with each iteration. When looking at validation accuracy over the epochs, we see that it is increasing, but not monotonically. This is because the network is generally getting better at modelling the data, but sometimes it might be over or under fitting thus performs worse on unseen data.

# 4 PyTorch Implementation

Test accuracy was 47.89% using the default settings. Very similar to the NumPy implementation, with a monotonically decreasing training loss and a validation accuracy curve that increases but not monotonically. Noteworthy is that the network seems to make a bigger jump in terms of training loss improvment after the first epoch and that it seems to over/underfit more, as the validation accuracy fluctuates and dips more considerably. This is further made evident by the slightly lower test accuracy.

# 5 Optimization

## 5.a Hessian Matrix at Local Minimum having Positive Eigenvalues

*Show that the Hessian matrix of a function $f(\mathbf{x})$ at a local minimum has only positive eigenvalues.*