Hand-In 2

Zhuyun Zhou 201801015
Jing Yin Ong 201800935

PART I: Derivative

$\text{softmax}(z)_j = \frac{e^{zj}}{\sum_{i=1}^{K} e^{zi}}$

$L(z) = -\lg(\text{softmax}(z)_j)$
Let $\text{softmax}(z)_j = p_j$
$L(z) = -\sum_i y_i \lg(p_i)$

$$\frac{\partial L}{\partial zi} = -\sum_k y_k \frac{\partial \log(\text{pk})}{\partial pk} \times \frac{\partial pk}{\partial zi}$$
$$= -\sum y_k \frac{1}{pk} \times \frac{\partial pk}{\partial zi}$$

$$\frac{\partial L}{\partial zi} = -y_i(1 - p_i) - \sum_{k \neq i} y_k \frac{1}{pk} (-p_k \cdot p_i)$$
$$= -y_i(1 - p_i) + \sum_{k \neq i} y_k \cdot p_i$$
$$= -y_i + y_i p_i + \sum_{k \neq i} y_k \cdot p_i$$
$$= p_i (y_i + \sum_{k \neq i} y_k) - y_i$$

y is an encoded vector for the labels, so $\sum_k y_k = 1$ and $y_i + \sum_{k \neq i} y_k = 1$
Thus,

$$\frac{\partial L}{\partial zi} = -y_i + p_i$$
$$= -\delta_{i,j} + \text{softmax}(z)_i$$

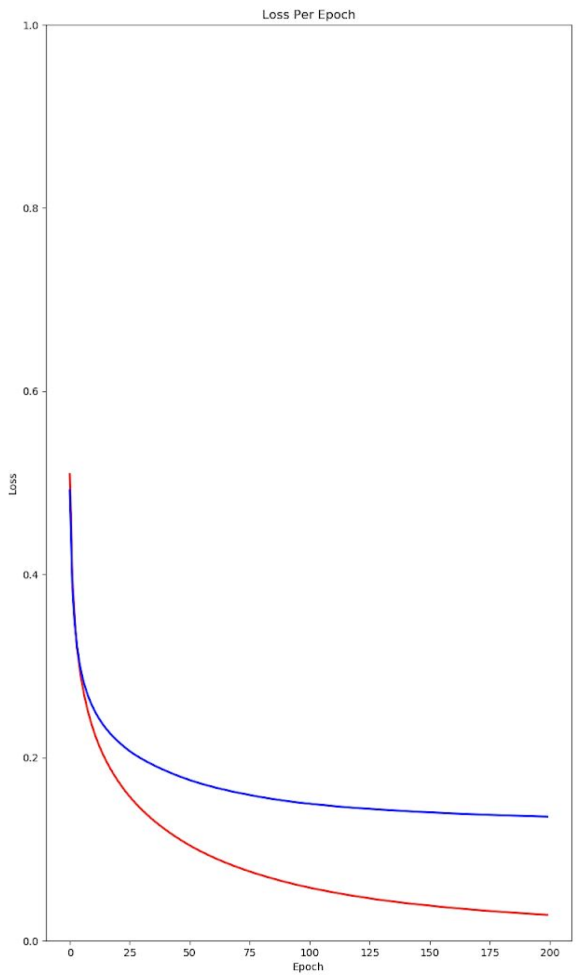
PART II: Implementation and Test

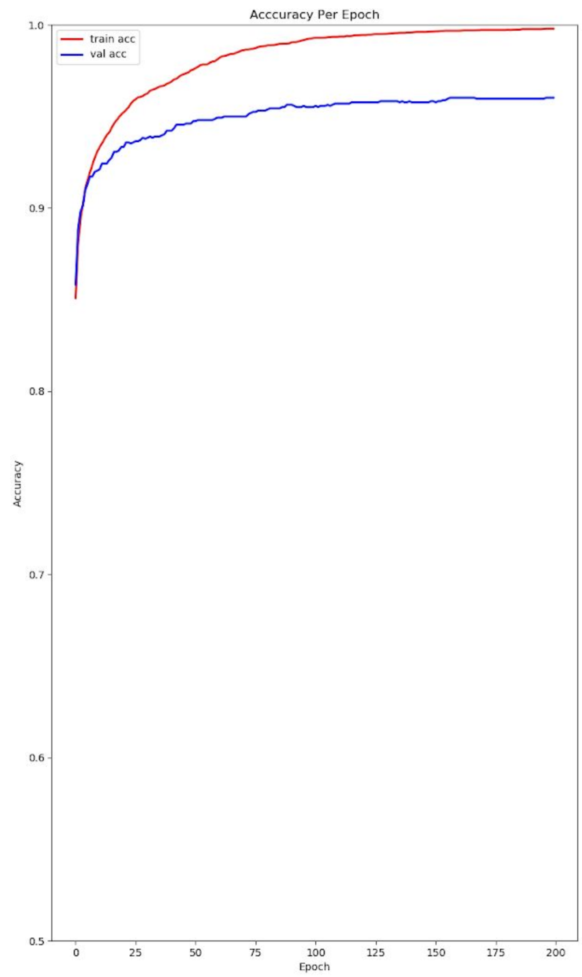
Forward and Backward pass

```
A1 = X @ W1 + b1
reluA1 = relu(A1)
A2 = reluA1 @ W2 + b2

log_softmax = np.log(softmax(A2))
entropy = np.sum(log_softmax * labels)
entropy = -entropy
decay = reg * (np.sum(np.square(W1)) + np.sum(np.square(W2)))

cost = entropy + decay
grad = -labels + softmax(A2)
d_b2 = grad.sum(axis=0, keepdims=True)
d_w2 = 2*reg*W2 + (reluA1.T @ grad)
d_RA1 = grad @ W2.T
d_A1 = d_RA1 * np.where(A1 > 0, 1, 0)
d_b1 = d_A1.sum(axis=0, keepdims=True)
d_w1 = X.T @ d_A1 + 2*reg*W1
```

Test

Using the parameters of **epochs = 200, batch_size = 30, lr = 0.0001, reg = 1e-2** for the fit function, we were able to achieve an in-sample accuracy and test sample accuracy of over 95%.






```
in sample accuracy 0.9921965317919075
data shape, type, min, max (2580, 784) float64 -1.0 1.0
labels shape and type (2580,) int64 0 9
test sample accuracy 0.951937984496124
outputting to file epoch_plots.png_
```