In solving the questions in this assignment, I worked together with my classmate Yi Yun Ding & 1004795214

I confirm that I have written the solutions/code/report in my own words.

Q1

Denote the output of a network as y

Denote the weight of ith layer as W_i

Denote the input of a network as x

(By hint)

$$y = b_n + W_n(b_{n-1} + W_{n-1}(\cdots(b_2 + W_2(b_1 + W_1 x))))$$

$$= b_n + W_n b_{n-1} + W_n W_{n-1} b_{n-2} + \cdots + W_n \cdots W_1 x$$

Denote all the terms with b_i as b'

Denote all the terms with $W_i x$ as $w'x$

Thus, $y = b' + w'x$

by this equation, we can easily find that adding layers does not increase the power of the linear neural network.

So we can say in a fully connected neural network with linear activation functions, the number of layers has effectively no impact on the network.

Q2:

By neural network architecture

$$S_1 = W_1x_1 + W_2x_2$$

$$S_2 = W_3x_3 + W_4x_4$$

$$\sigma_1 = \frac{1}{1+e^{-S_1}} \qquad \sigma_2 = \frac{1}{1+e^{-S_2}}$$

$$S_3 = W_5\sigma_1 + W_6\sigma_2$$

$$\hat{y} = \sigma_3 = \frac{1}{1+e^{-S_3}}$$

By Chain Rule

$$\frac{\partial L}{\partial W_3} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial S_3} \frac{\partial S_3}{\partial \sigma_2} \frac{\partial \sigma_2}{\partial S_2} \frac{\partial S_2}{\partial W_3}$$

$$\frac{\partial L}{\partial \hat{y}} = -2\,\|y - \hat{y}\| \qquad \frac{\partial \sigma_2}{\partial S_2} = \frac{e^{-S_2}}{(e^{-S_2}+1)^2}$$

$$\frac{\partial \hat{y}}{\partial S_3} = \frac{e^{-S_3}}{(e^{-S_3}+1)^2} \qquad \frac{\partial S_2}{\partial W_3} = x_3$$

$$\frac{\partial S_3}{\partial \sigma_2} = W_6$$

$y = 0.5 \qquad W_6 = -0.2 \qquad x_3 = -0.3$

$$S_2 = W_3x_3 + W_4x_4 = 0.24 \times (-0.3) + (-1.7) \times 0.8$$
$$= -0.072 - 1.36 = -1.432$$

$$S_1 = W_1x_1 + W_2x_2 = 0.75 \times 0.9 + (-0.63) \times (-1.1)$$
$$= 1.368$$

$$\sigma_1 = \frac{1}{1+e^{-s_1}} \approx 0.7971 \qquad \sigma_2 = \frac{1}{1+e^{-s_2}} \approx 0.1928$$

$$\begin{aligned} s_3 = w_5\,\sigma_1 + w_6\,\sigma_2 &= 0.8 \times 0.7971 + (-0.2) \times 0.1928 \\ &= 0.63768 - 0.03856 \\ &= 0.59912 \\ &\approx 0.5991 \end{aligned}$$

$$\hat{y} = \frac{1}{1+e^{-s_3}} \approx 0.6455$$

$$\Rightarrow \frac{\partial L}{\partial \hat{y}} = 2\,|0.5 - 0.6455| = 0.291$$

$$\frac{\partial \hat{y}}{\partial s_3} = \frac{e^{-0.5991}}{(e^{-0.5991}+1)^2} \approx \frac{2.1192}{(2.1192+1)^2} \approx \frac{2.1192}{9.7294} \approx 0.2178$$

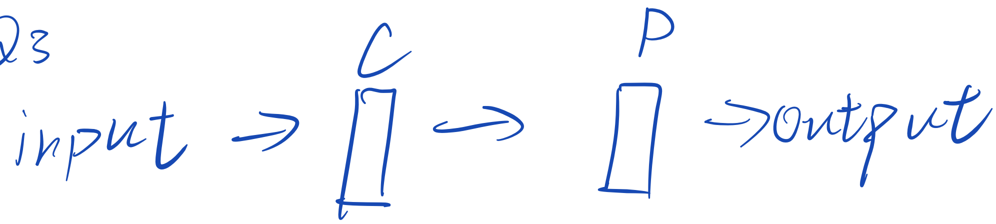
$$\frac{\partial s_3}{\partial \sigma_2} = -0.2 \qquad \frac{\partial s_2}{\partial w_2} = -0.3$$

$$\frac{\partial \sigma_2}{\partial s_2} = \frac{e^{1.432}}{(e^{1.432}+1)^2} \approx \frac{4.1871}{(4.1871+1)^2} \approx \frac{4.1871}{26.9056} \approx 0.1556$$

$$\Rightarrow \frac{\partial L}{\partial w_3} = 0.241 \times 0.2178 \times (-0.2) \times (-0.3) \times 0.1556$$

$$\approx 0.0006$$

Q3



For the layer C (convolution)

Let F_c denote the number of flops during convolution

$$F_c = [(C_i \times K_w \times K_h) + (C_i \times K_w \times K_h - 1) + 1] \times C_o \times W \times H$$

because of bias

C_i – Channel
K_w – kernel width
K_h – kernel height

C_o – output channels
W – output width
H – output height

output shape = (input size − filter size + padding×2)/stride + 1

$$= (12 - 4 + 2)/2 + 1$$
$$= 6$$

$$F_c = [(50 \times 4 \times 4) + (50 \times 4 \times 4 - 1)] \times 20 \times 6 \times 6 + 1 \times 20 \times 6 \times 6$$
$$= 1151280 + 720$$ ← biases

Let F_p denote number of flops during max pooling

Max pooling horizontally would be $(6-3) \div 1 + 1 = 4$ times

vertically would also be $(6-3) \div 1 + 1 = 4$ times

$$F_p = (3 \times 3 - 1) \times (4 \times 4) \times 20$$
$$= 2560$$

So in total

with bias: $F = 1151280 + 720 + 2560$
$= 11545600$

without bias: $F = 1151280 + 2560$
$= 1153840$

Q4

Let P_c denotes the number of parameters in convolution layer

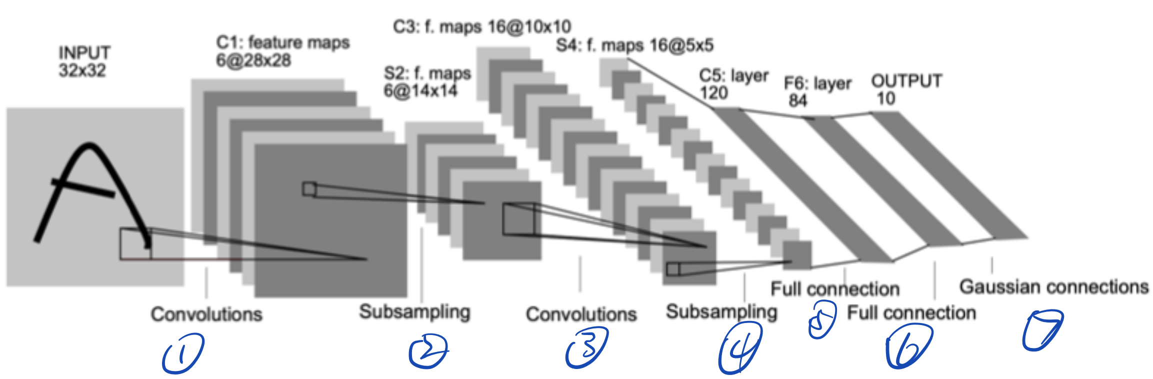
P_c = ((shape of width of filter × shape of height of filter × number of filters in previous layer + 1) × number of filters

bias

Let P_f denotes the number of parameters in fully connected layer

P_f = (current layer neurons c × previous layer neurons p) + c

For subsampling, parameter number is 0.



① : $(5\times5\times1+1)\times6=156$ ② : 0

③ : $(5\times5\times6+1)\times16=2416$ ④ : 0

⑤ : $(16\times5\times5+1)\times120=48120$

⑥ : $(120+1)\times84=10164$ ⑦ $(84+1)\times10=850$

Total: $156+2416+48120+10164+850=61706$

Q5

For Logistic activation function

$$f(x) = \frac{1}{1+e^{-x}}$$

$$f'(x) = \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}}$$

$$= \frac{1}{1+e^{-x}} \cdot \left(1 - \frac{1}{1+e^{-x}}\right)$$

$$= f(x) \cdot (1 - f(x))$$

So if we have output of neurons, there is no need for the inputs.

Q6

a) hyperbolic tangent function range: $(-1, 1)$

logistic function: $(0, 1)$

b) Let $\sigma(x)$ denote logistic function

$$\sigma(x) = \frac{1}{1-e^{-x}} \Rightarrow \sigma(2x) = \frac{1}{1-e^{-2x}}$$

by property $1-\sigma(x) = \sigma(-x)$

$$\tanh(x) = \frac{1-e^{-2x}}{1+e^{-2x}} = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} = \frac{e^{x}+e^{-x}-2e^{-x}}{e^{x}+e^{-x}}$$

$$= 1 + \frac{-2e^{-x}}{e^{x}+e^{-x}}$$

$$= 1 - \frac{2}{e^{2x}+1}$$

$$= 1 - 2\sigma(-2x)$$

$$= 1 - 2(1-\sigma(2x))$$

$$= 1 - 2 + 2\sigma(2x)$$

$$= 2\sigma(2x) - 1$$

$$\frac{\partial \tanh(x)}{\partial x} = 2\sigma'(2x) \cdot 2 = 4\sigma'(2x)$$

$$\sigma'(x) = \sigma(x)(1-\sigma(x))$$

$$\Rightarrow \sigma'(2x) = \sigma(2x)(1-\sigma(2x))$$

$$\Rightarrow \frac{\partial \tanh(x)}{\partial x} = 4\,\sigma(2x)(1-\sigma(2x))$$

c) For logistic activation function, it is used for models that need to predict probability since it's range is (0, 1).

For hyperbolic tanh function, it is mainly used classification between two classes.

Task II

Learn rate: 0.0002

Training error: 1 - 95.22% = 4.78%

Validation error: 1 – 92.26% = 7.74%

Test Error: 1 – 89.60% = 10.40%

```
Text(0.5, 1.0, 'Training, Validation and Test error')
```

Learn rate: 0.0025

Training error: 1 – 93.05% = 6.95%

Validation Error: 1 – 91.16% = 8.84%

Test error: 1 – 90.00% = 10.00%

```
Text(0.5, 1.0, 'Training, Validation and Test error')
```

Learn rate: 0.002

Training error: 1 – 84.26% = 15.74%

Validation Error: 1 – 81.91% = 18.09%

Test error: 1 – 81.00% = 19.00%

```
Text(0.5, 1.0, 'Training, Validation and Test error')
```

Learn rate: 0.003

Training error: 1 – 91.59% = 8.41%

Validation Error: 1 – 90.68% = 9.32%

Test error: 1 – 89.40% = 10.60%

```
Text(0.5, 1.0, 'Training, Validation and Test error')
```

Learn rate: 0.0001

Training error: 1 – 94.35% = 5.65%

Validation Error: 1 – 91.60% = 8.40%

Test error: 1 – 89.70% = 10.30%

```
Text(0.5, 1.0, 'Training, Validation and Test error')
```

In conclusion, when learn rate is 0.0001, the validation error is least.

Task III

Learn rate: 0.0002

Hidden Units: 100

Validation Error: 1 – 91.23% = 8.77%

Test error: 1 – 89.60% = 10.40%

Hidden Units: 500

Validation Error: 1 – 92.11% = 7.89%

Test error: 1 – 89.50% = 10.50%

Hidden Units: 1000

Validation error: 1 – 92.26% = 7.74%

Test Error: 1 – 89.60% = 10.40%

When the number of hidden units increases, the validation error decreases.

Task IV

Learn rate: 0.0002

One hidden layer

Validation error: 1 – 92.26% = 7.74%

```
Text(0.5, 1.0, 'Training, Validation and Test error')
```

Two hidden layers

Validation Error: 1 – 83.20% = 16.80%

```
Text(0.5, 1.0, 'Training, Validation and Test error')
```

Compare two architecture, one layer case has more training parameters than two-layer cases, so the validation accuracy would increase, and the validation error would decrease.

Task V

Learn rate: 0.0002

Without Dropout

Training error: 1 - 95.22% = 4.78%

Validation error: 1 – 92.26% = 7.74%

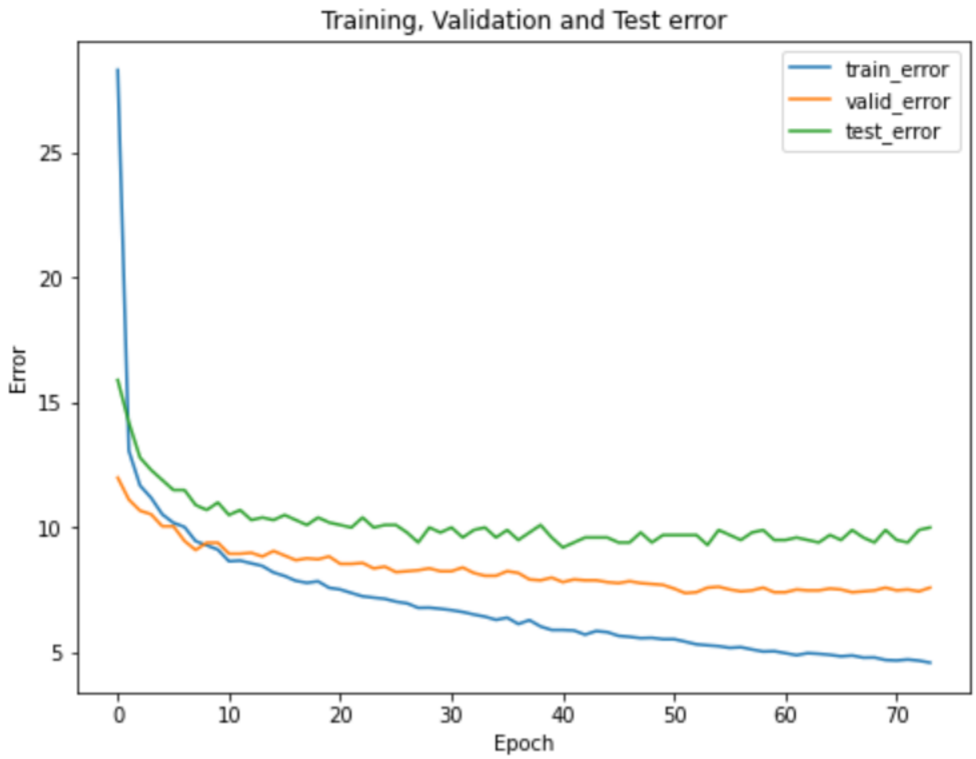
```
Text(0.5, 1.0, 'Training, Validation and Test error')
```

With Dropout

Training error: 1 - 95.41% = 4.59%

Validation error: 1 – 92.41% = 7.59%

```
Text(0.5, 1.0, 'Training, Validation and Test error')
```



With dropout, the training, validation accuracy all increases, training and validation error decreases compared with the accuracy without dropout.