Q1

$$\varphi'(s) = s\,\theta'(s) + \theta(s)$$

$$\frac{d\theta(s)}{ds} = \left[(1+e^{-s})^{-1}\right]' = -e^{-s}\cdot(-1)\cdot(1+e^{-s})^{-2}$$

$$= \frac{e^{-s}}{(1+e^{-s})^2} = (1-\theta(s))\,\theta(s)$$

$$\Rightarrow \varphi'(s) = (s+1)\,\theta(s) - s(\theta(s))^2$$

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

Q2

(A) $V_0 = \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]^T$

$$V_1 = \begin{bmatrix} 0 & 1 & 1/2 \\ 0 & 0 & 1/2 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/6 \\ 1/3 \end{bmatrix}$$

$$V_2 = \begin{bmatrix} 0 & 1 & 1/2 \\ 0 & 0 & 1/2 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1/2 \\ 1/6 \\ 1/3 \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/6 \\ 1/2 \end{bmatrix}$$

$$V_3 = \begin{bmatrix} 0 & 1 & 1/2 \\ 0 & 0 & 1/2 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/6 \\ 1/2 \end{bmatrix} = \begin{bmatrix} 5/12 \\ 1/4 \\ 1/3 \end{bmatrix}$$

$$V_4 = \begin{bmatrix} 0 & 1 & 1/2 \\ 0 & 0 & 1/2 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 5/12 \\ 1/4 \\ 1/3 \end{bmatrix} = \begin{bmatrix} 5/12 \\ 1/6 \\ 5/12 \end{bmatrix}$$

$$V_5 = \begin{bmatrix} 0 & 1 & 1/2 \\ 0 & 0 & 1/2 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 5/12 \\ 1/6 \\ 5/12 \end{bmatrix} = \begin{bmatrix} 3/8 \\ 5/24 \\ 5/12 \end{bmatrix}$$

(B)

Suppose $V^* = (a, b, c)$, $a+b+c = 1$

$$\Rightarrow \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1/2 \\ 0 & 0 & 1/2 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} \Rightarrow \begin{array}{l} a = b + \frac{1}{2}c \\ b = \frac{1}{2}c \\ c = a \end{array} \Rightarrow \frac{5}{2}a = 1$$

$$\Rightarrow V^* = \left[\frac{2}{5}, \frac{1}{5}, \frac{2}{5}\right]^T \#$$

3. Let $d_{(\ell)} = d^{(\ell)}$

If $L = 1$

$\Rightarrow d_0 + d_1 = 100$, $d_1 = 90 \Rightarrow$ the only possible number of weights is 900

If $L = 2$

$\Rightarrow d_0 + d_1 + d_2 = 100 \Rightarrow d_1 + d_2 = 90$ $\quad \nearrow \begin{pmatrix} d_1 = 50 \\ 2d_1 = 100 \end{pmatrix}$

$W = 10d_1 + d_1 d_2 = 10d_1 + d_1(90 - d_1) = -d_1^2 + 100d_1$

$\Rightarrow (d_1, d_2) = (50, 40)$ has maxi. number of weights $= 2500$

$\Rightarrow (d_1, d_2) = (1, 89)$ has mini. number of weights $= 99$

If $L = 3$

$\Rightarrow d_1 + d_2 + d_3 = 90 = g(d_1, d_2, d_3)$

$$\begin{aligned} W(d_1, d_2, d_3) &= 10d_1 + d_1 d_2 + d_2 d_3 \\ &= 10(90 - d_2 - d_3) + (90 - d_2 - d_3) d_2 + d_2 d_3 \\ &= 900 - 10d_2 - 10d_3 + 90d_2 - d_2^2 \\ &= 900 + 80d_2 - d_2^2 - 10d_3 \\ &= -(d_2 - 40)^2 + 2500 - 10d_3 \end{aligned}$$

$\Rightarrow (d_1, d_2, d_3) = (49, 40, 1)$ has max. number of weights $= 2490$

$\Rightarrow (d_1, d_2, d_3) = (1, 1, 88)$ has mini. number of weights $= 99$

Hence, the maximum number of weights is 2500,

$L=2$, $(d_0, d_1, d_2) = (10, 50, 40)$

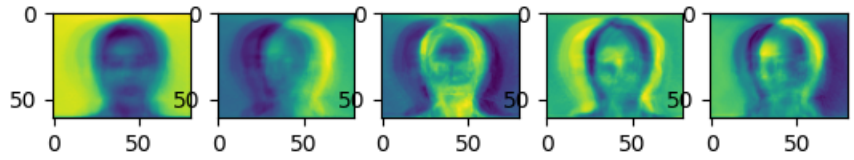
the minimum number of weights is 99,

$L=2$, $(d_0, d_1, d_2) = (10, 1, 89)$ or

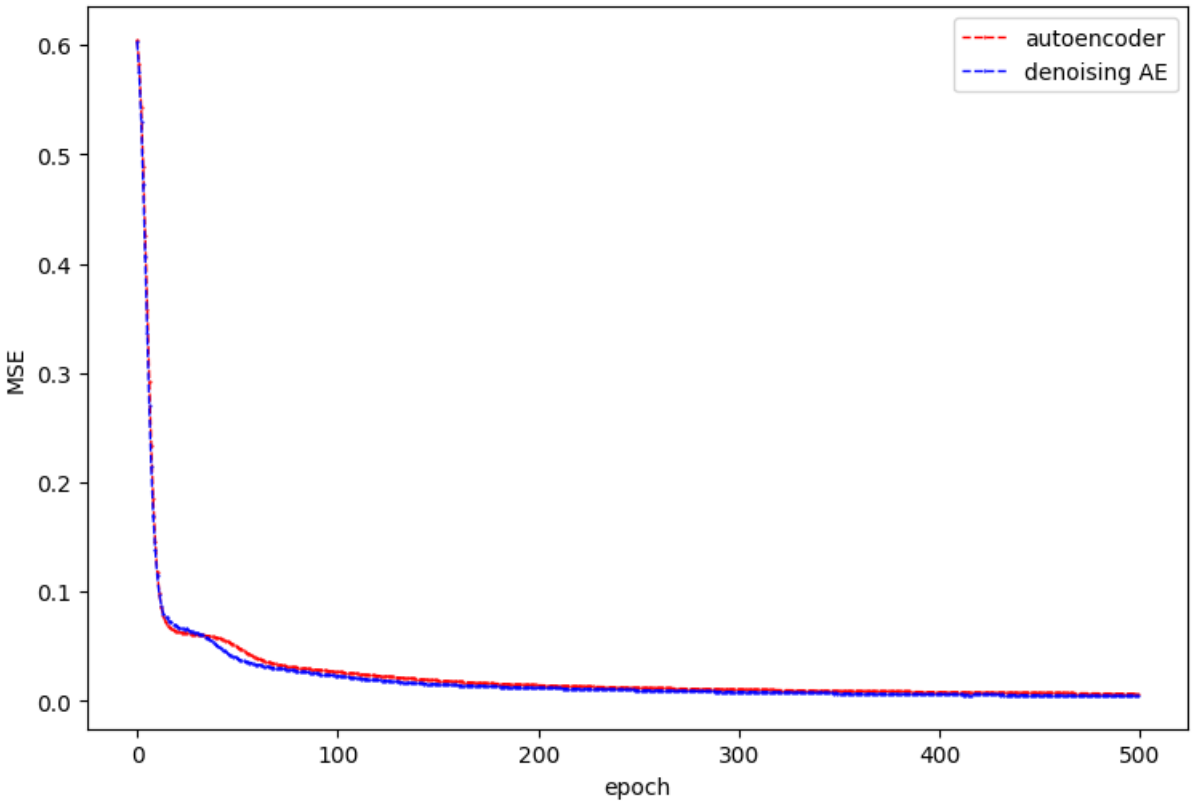
$L=3$, $(d_0, d_1, d_2, d_3) = (10, 1, 1, 88)$ #

Programming Part Report

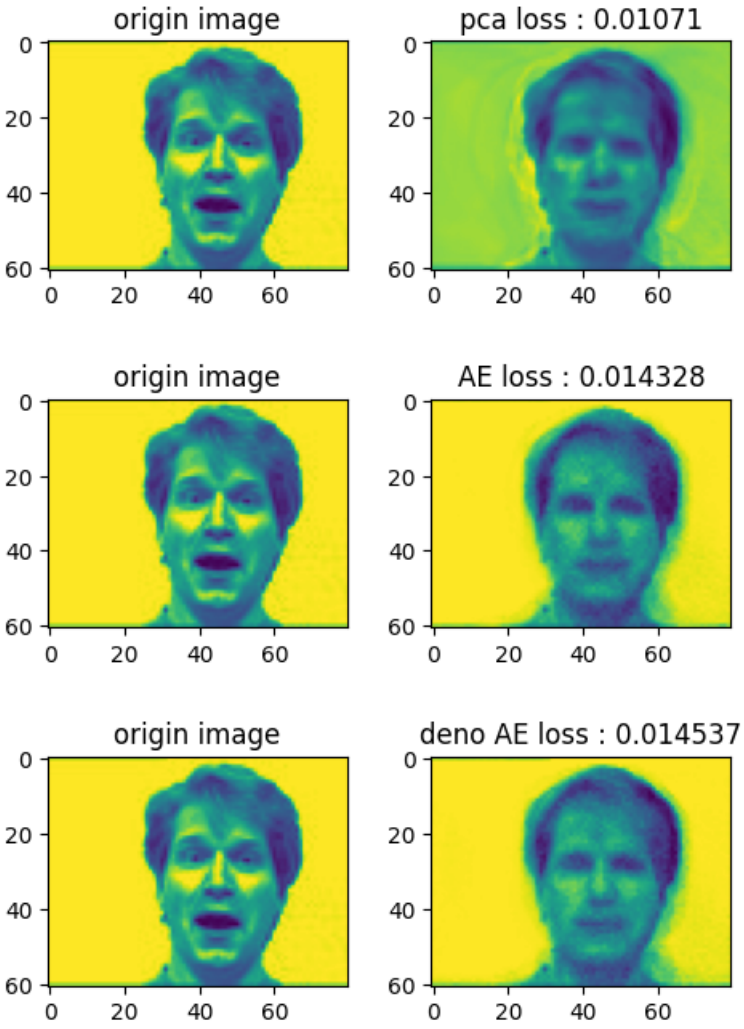
(a)



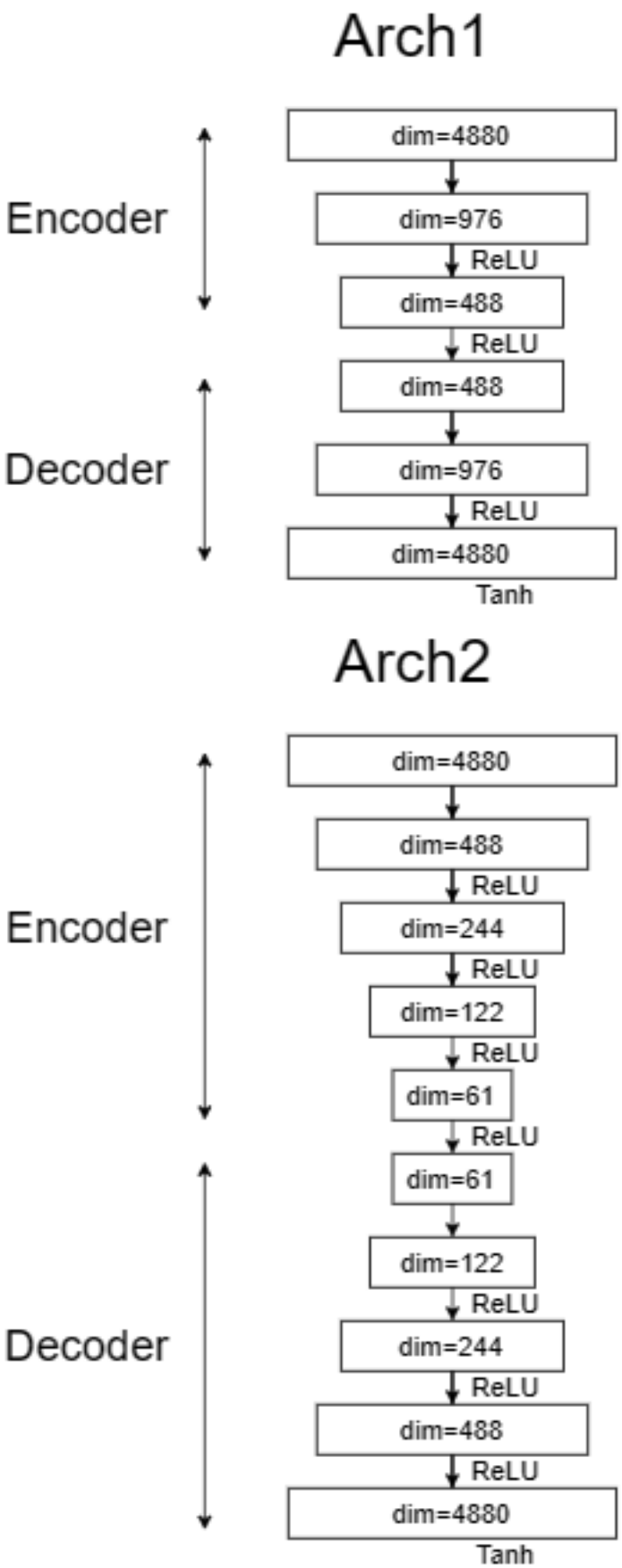
(b)

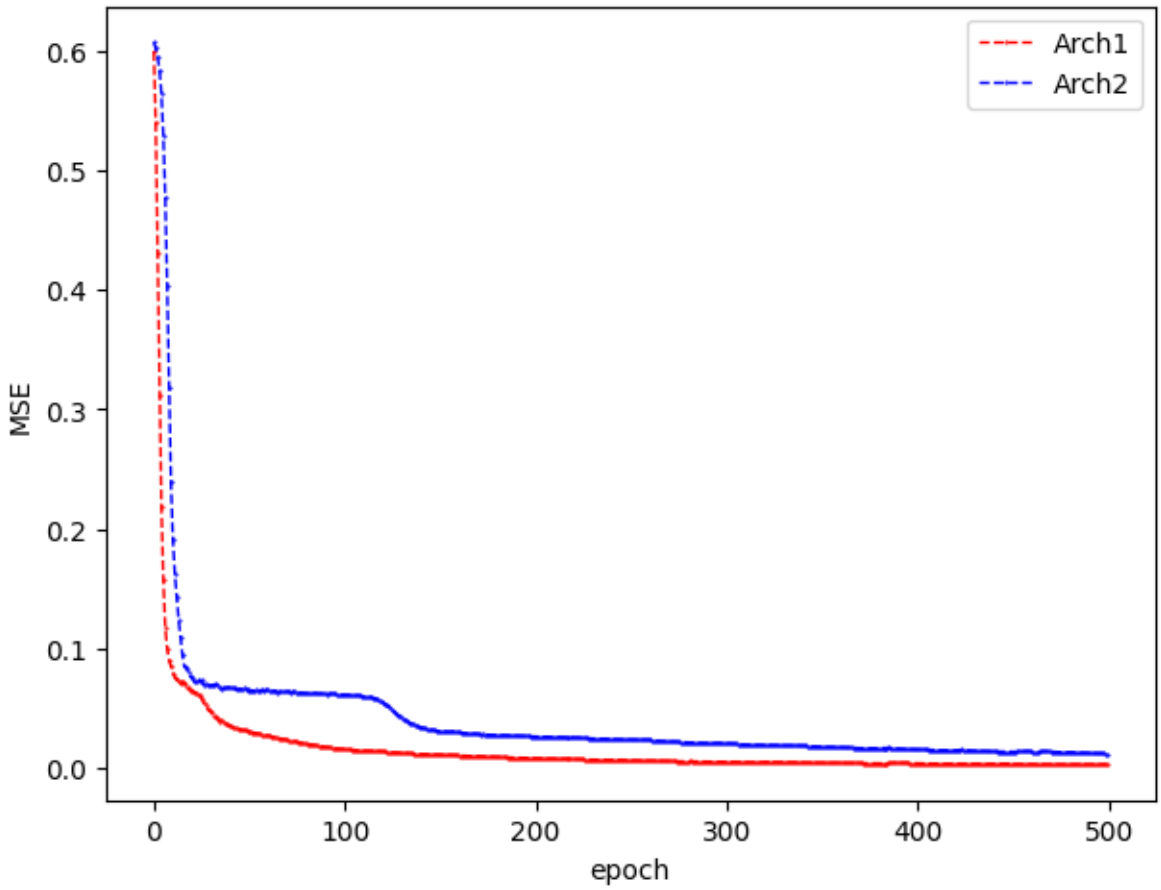

autoencoder
denoising AE
MSE
epoch
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0
100
200
300
400
500

(c)


origin image
pca loss : 0.01071
origin image
AE loss : 0.014328
origin image
deno AE loss : 0.014537

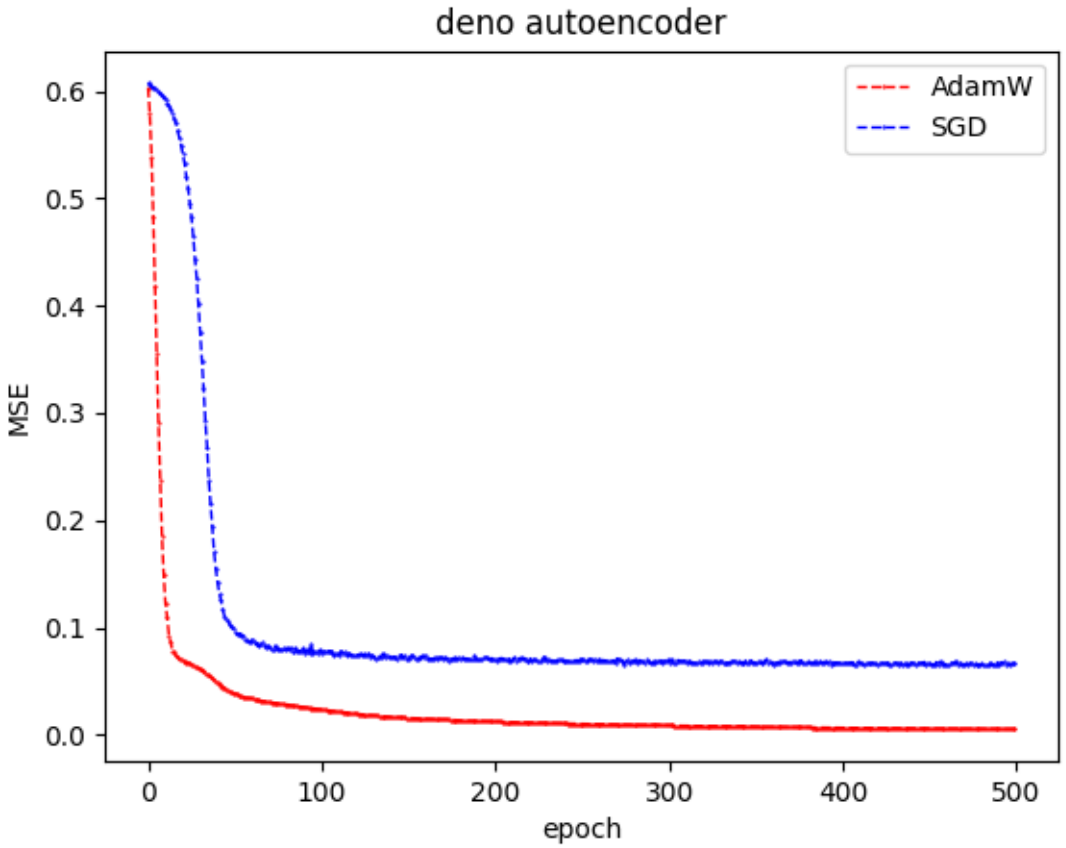
(d)


Arch1
Encoder
Decoder
dim=4880
dim=976
ReLU
dim=488
ReLU
dim=488
dim=976
ReLU
dim=4880
Tanh
Arch2
Encoder
Decoder
dim=4880
dim=488
ReLU
dim=244
ReLU
dim=122
ReLU
dim=61
ReLU
dim=61
dim=122
ReLU
dim=244
ReLU
dim=488
ReLU
dim=4880
Tanh



The two architectures are trained under epoch:500, batchSize:45, learning rates:1e-4 and optimizer:AdamW. Observing the trainning curve, I find the curve of Arch1 is generally smaller than the curve of Arch2, that is, the MSE loss of Arch1 is smaller than the one of Arch2. Also, when it comes to reconstruction loss, the loss of Arch1 is about 0.01413 while the loss of Arch2 is about 0.01783. Based on these results, I think a shallow but fatter denoising autoencoder may outperform a deeper but thinner one. This might because a fatter network can preserve more information when decoding and a deeper network may be more difficult to optimize.

(e)



I tired two optimizer, one is vanilla SGD and another one is AdamW. Based on the trainning curve, I find AdamW can converge slightly faster than SGD. However, the overall performence of AdamW defeat the one of SGD totally. The result may because AdamW can dynamically adjust learning rates and use momentum to improve trainning while vanilla SGD simply train the model.