人工智慧導論 HW4

許博翔

June 5, 2024

Reference: all by myself

1 Hand-written Part

Problem 1. $\varphi'(s) = \frac{1 \cdot (1+e^{-s}) - s \cdot (-e^{-s})}{(1+e^{-s})^2} = \frac{1+(1+s)e^{-s}}{(1+e^{-s})^2}$.

Problem 2.

(A) $\mathbf{v}_0 = \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{pmatrix}$.

$$\mathbf{v}_1 = \begin{pmatrix} 0 & 1 & 0.5 \\ 0 & 0 & 0.5 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{6} \\ \frac{1}{3} \end{pmatrix}.$$

$$\mathbf{v}_2 = \begin{pmatrix} 0 & 1 & 0.5 \\ 0 & 0 & 0.5 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{2} \\ \frac{1}{6} \\ \frac{1}{3} \end{pmatrix} = \begin{pmatrix} \frac{1}{3} \\ \frac{1}{6} \\ \frac{1}{2} \end{pmatrix}.$$

$$\mathbf{v}_3 = \begin{pmatrix} 0 & 1 & 0.5 \\ 0 & 0 & 0.5 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{3} \\ \frac{1}{6} \\ \frac{1}{2} \end{pmatrix} = \begin{pmatrix} \frac{5}{12} \\ \frac{1}{4} \\ \frac{1}{3} \end{pmatrix}.$$

$$\mathbf{v}_4 = \begin{pmatrix} 0 & 1 & 0.5 \\ 0 & 0 & 0.5 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{5}{12} \\ \frac{1}{4} \\ \frac{1}{3} \end{pmatrix} = \begin{pmatrix} \frac{5}{12} \\ \frac{1}{6} \\ \frac{5}{12} \end{pmatrix}.$$

$$\mathbf{v}_5 = \begin{pmatrix} 0 & 1 & 0.5 \\ 0 & 0 & 0.5 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{5}{12} \\ \frac{1}{6} \\ \frac{5}{12} \end{pmatrix} = \begin{pmatrix} \frac{3}{8} \\ \frac{5}{24} \\ \frac{5}{12} \end{pmatrix}.$$

(B) Suppose that $\mathbf{v}^* = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$.

$$\mathbf{v}^* = \begin{pmatrix} 0 & 1 & 0.5 \\ 0 & 0 & 0.5 \\ 1 & 0 & 0 \end{pmatrix} \mathbf{v}^*.$$

$$\Rightarrow \left(\begin{pmatrix} 0 & 1 & 0.5 \\ 0 & 0 & 0.5 \\ 1 & 0 & 0 \end{pmatrix} - I \right) \mathbf{v}^* = 0.$$

$$\Rightarrow \begin{pmatrix} -1 & 1 & 0.5 \\ 0 & -1 & 0.5 \\ 1 & 0 & -1 \end{pmatrix} \mathbf{v}^* = 0.$$

$\Rightarrow -v_2 + 0.5v_3 = 0, v_1 - v_3 = 0.$

$$\Rightarrow \mathbf{v}^* = \begin{pmatrix} v_3 \\ 0.5v_3 \\ v_3 \end{pmatrix}.$$

Since $(1 + 0.5 + 1)v_3 = 1$, there is $v_3 = 0.4$.

$$\therefore \mathbf{v}^* = \begin{pmatrix} 0.4 \\ 0.2 \\ 0.4 \end{pmatrix}.$$

Problem 3.

(A)
- Iteration 1:

  Partition: $\{(1,2)\}, \{(3,4), (7,0), (10,2)\}$

  Centroids: $(1,2), (\frac{20}{3}, 2)$

- Iteration 2:

  Partition: $\{(1,2), (3,4)\}, \{(7,0), (10,2)\}$

  Centroids: $(2,3), (8.5,1)$

- Iteration 3:

  Partition: $\{(1,2),(3,4)\},\{(7,0),(10,2)\}$

  Centroids: $(2,3),(8.5,1)$

The convergent result:

Partition: $\{(1,2),(3,4)\},\{(7,0),(10,2)\}$

Centroids: $(2,3),(8.5,1)$

(B)
- Iteration 1:

  Partition: $\{(1,2),(3,4)\},\{(7,0),(10,2)\}$

  Centroids: $(2,3),(8.5,1)$

- Iteration 2:

  Partition: $\{(1,2),(3,4)\},\{(7,0),(10,2)\}$

  Centroids: $(2,3),(8.5,1)$

The convergent result:

Partition: $\{(1,2),(3,4)\},\{(7,0),(10,2)\}$

Centroids: $(2,3),(8.5,1)$

The result is same to (A).

(C) Consider the two different initial centroids: $\{(1,2),(3,4)\}$ and $\{(1,2),(7,0)\}$.

- The former centroids:

  - Iteration 1:

    Partition: $\{(1,2)\},\{(3,4),(5,6),(7,0),(10,2)\}$

    Centroids: $(1,2),(6.25,3)$

  - Iteration 2:

    Partition: $\{(1,2),(3,4)\},\{(5,6),(7,0),(10,2)\}$

    Centroids: $(2,3),(\frac{22}{3},\frac{7}{3})$

  - Iteration 3:

    Partition: $\{(1,2),(3,4)\},\{(5,6),(7,0),(10,2)\}$

    Centroids: $(2,3),(\frac{22}{3},\frac{7}{3})$

The convergent result:

Partition: $\{(1,2),(3,4)\},\{(5,6),(7,0),(10,2)\}$

Centroids: $(2,3),(\frac{22}{3},\frac{7}{3})$

$\mathrm{E_{in}} = \frac{53}{3}$.

- The latter centroids:
  - Iteration 1:

    Partition: $\{(1,2),(3,4),(5,6)\},\{(7,0),(10,2)\}$

    Centroids: $(3,4),(8.5,1)$
  - Iteration 2:

    Partition: $\{(1,2),(3,4),(5,6)\},\{(7,0),(10,2)\}$

    Centroids: $(3,4),(8.5,1)$

  The convergent result:

  Partition: $\{(1,2),(3,4),(5,6)\},\{(7,0),(10,2)\}$

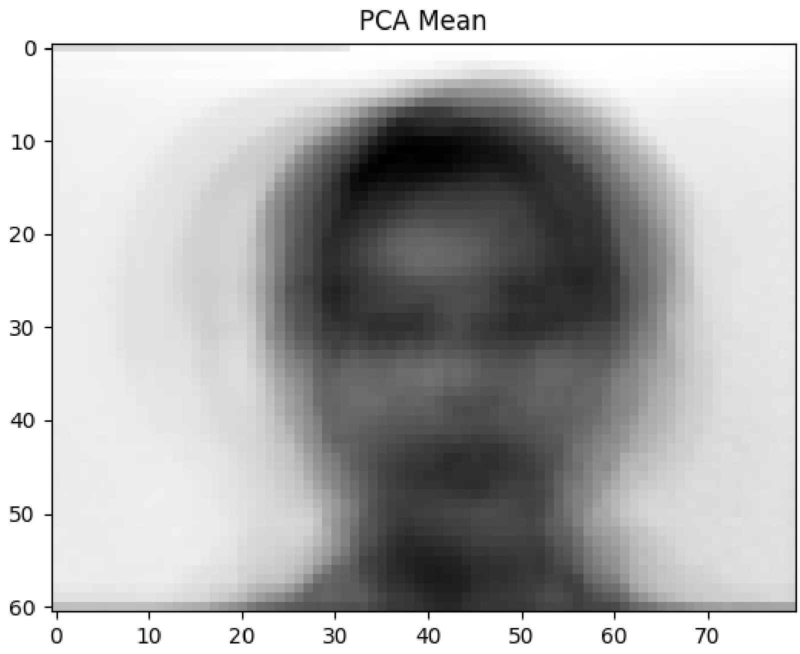
  Centroids: $(3,4),(8.5,1)$

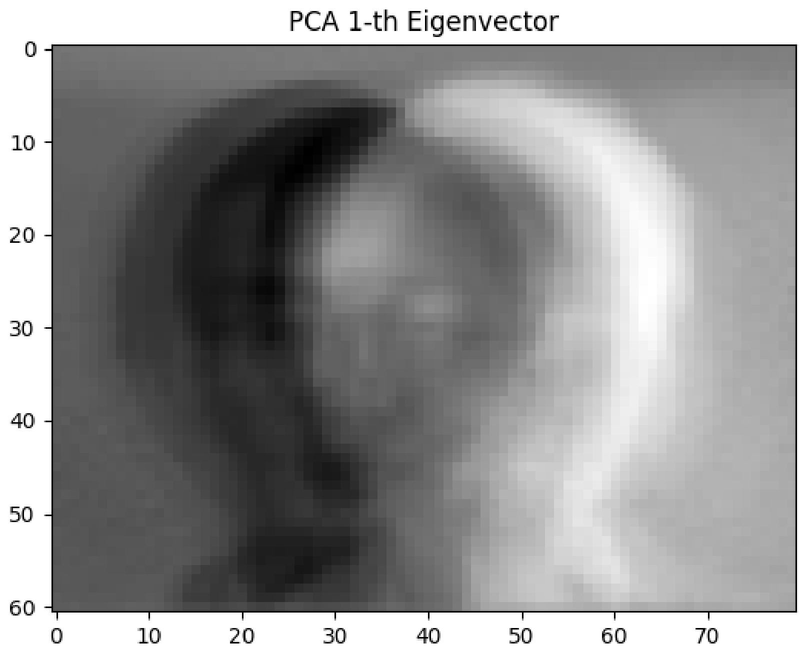
  $\mathrm{E_{in}} = 11.25$.

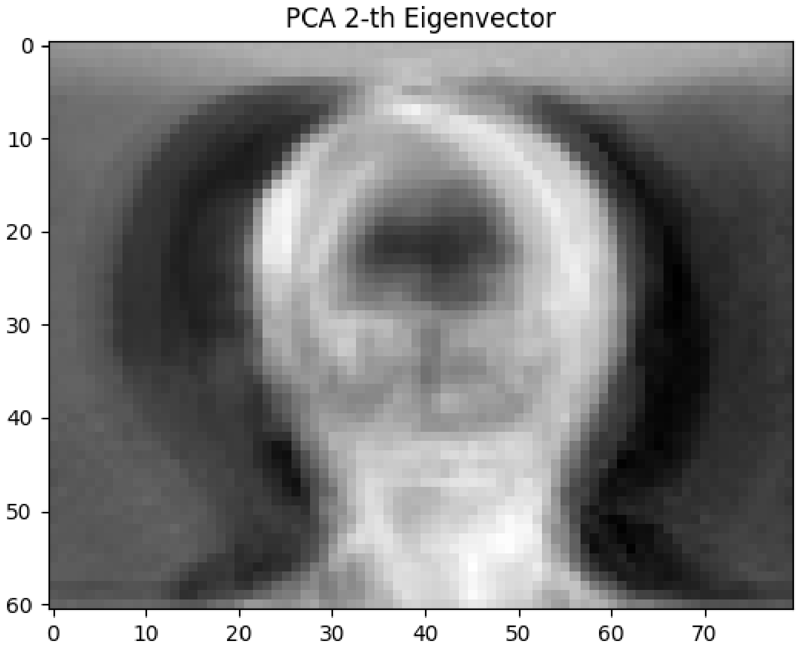
Since $11.25 \neq \frac{53}{3} \approx 17.67$, at least one of the above does not converge to the global minimum.

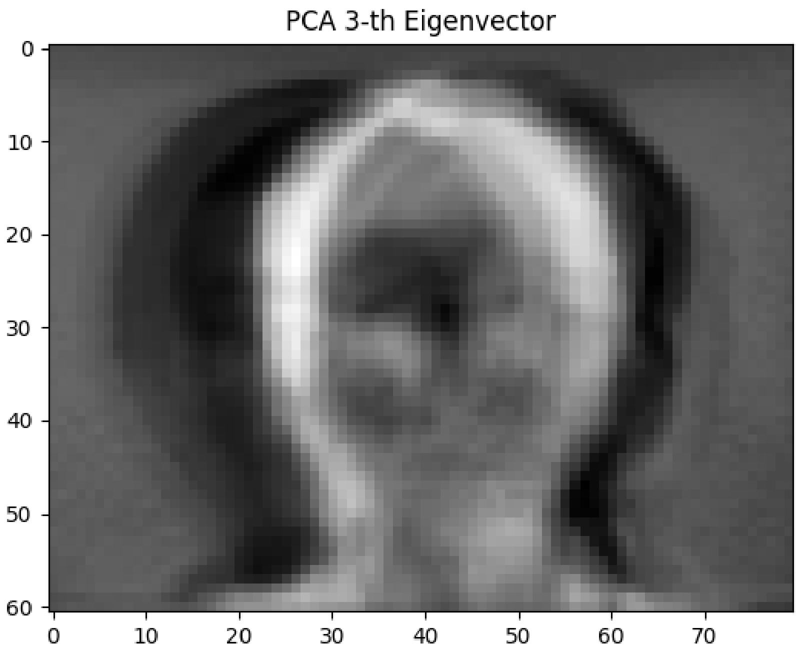
2 Programming Part

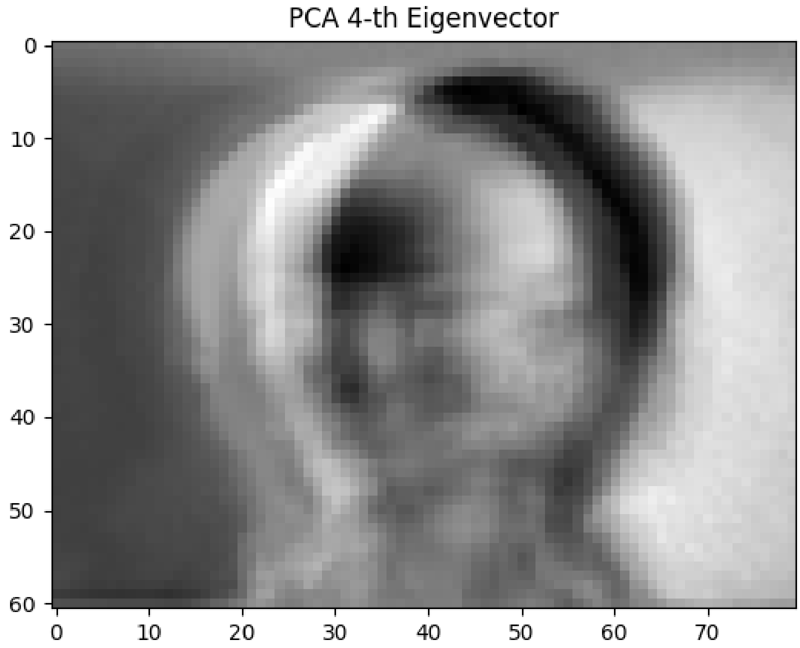
2.1 (a)



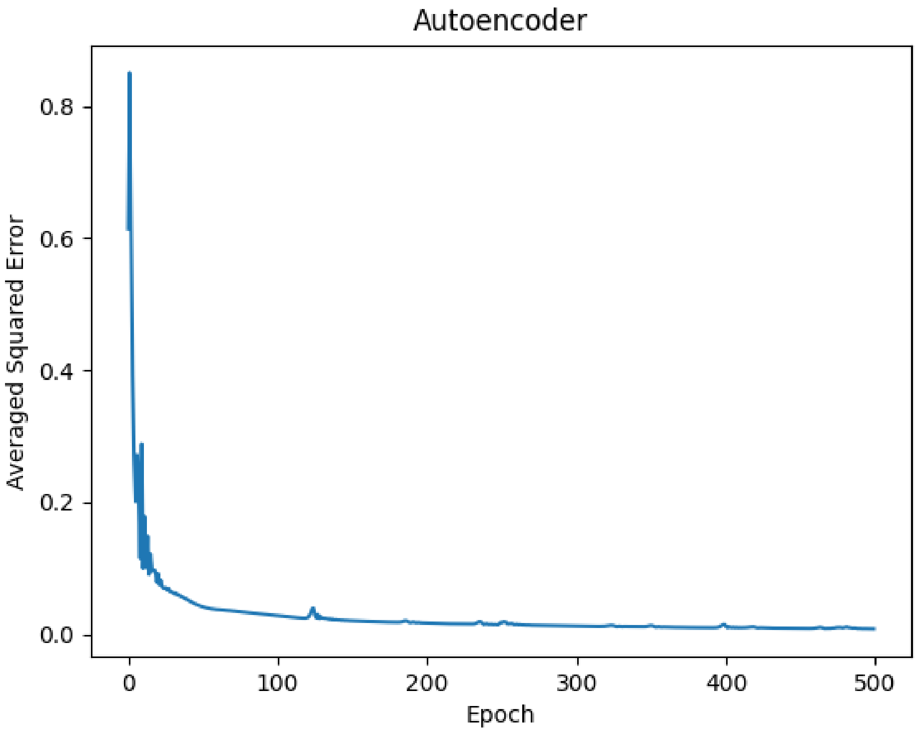


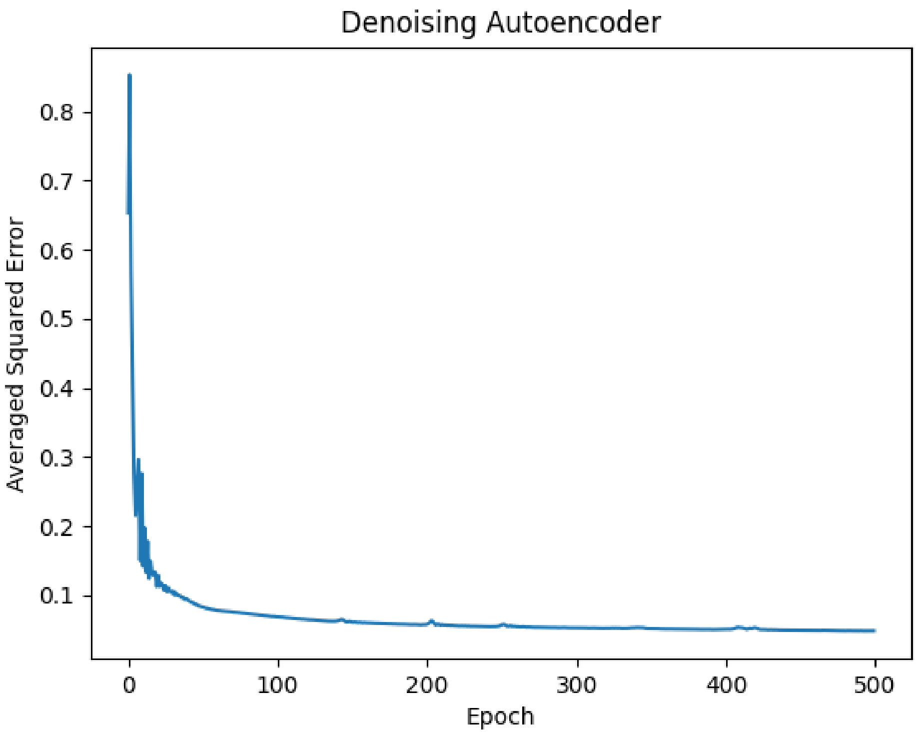

PCA 2-th Eigenvector
0
10
20
30
40
50
60
0 10 20 30 40 50 60 70


PCA 3-th Eigenvector
0
10
20
30
40
50
60
0 10 20 30 40 50 60 70


PCA 4-th Eigenvector
0
10
20
30
40
50
60
0
10
20
30
40
50
60
70

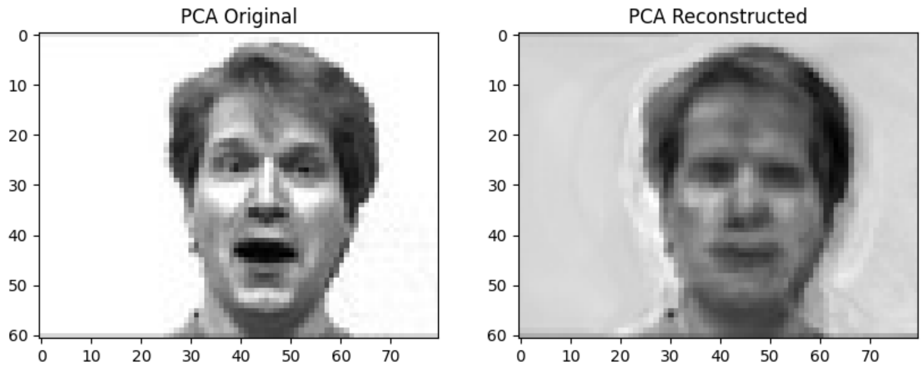
2.2 (b)


Autoencoder
Averaged Squared Error
0.0
0.2
0.4
0.6
0.8
0
100
200
300
400
500
Epoch

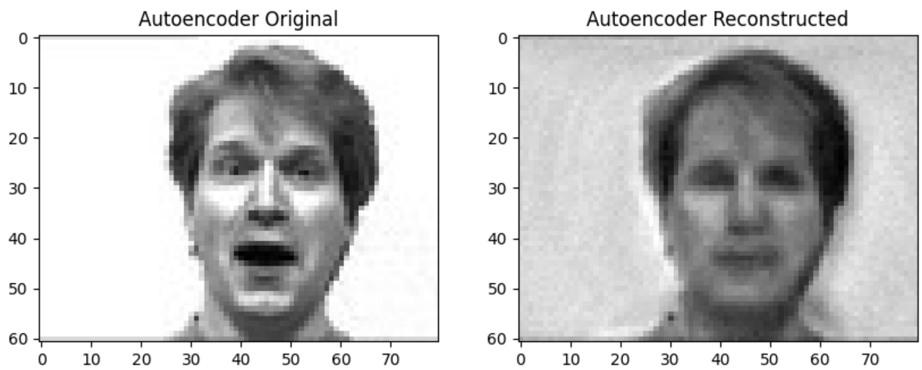

Denoising Autoencoder
Averaged Squared Error
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
100
200
300
400
500
Epoch

2.3 (c)

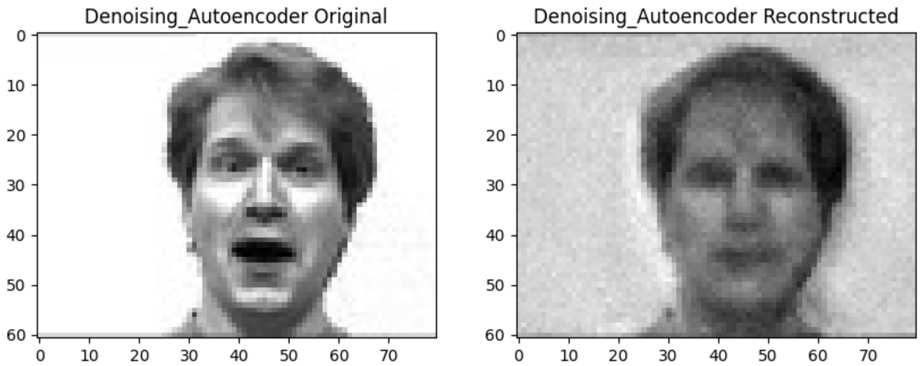
PCA mean squared error: 0.010710469688056315.



Autoencoder mean squared error: 0.012823789826138236.



DenoisingAutoencoder mean squared error: 0.01361696472773301.



2.4 (d)

Architecture 1:

4880 neorons → 488 neorons → 4880 neorons

Mean squared error: 0.019768476466042514.

Architecture 2:

4880 neorons → 488 neorons → 244 neorons → 488 neorons → 4880 neorons

Mean squared error: 0.013616964727773301.

Architecture 3:

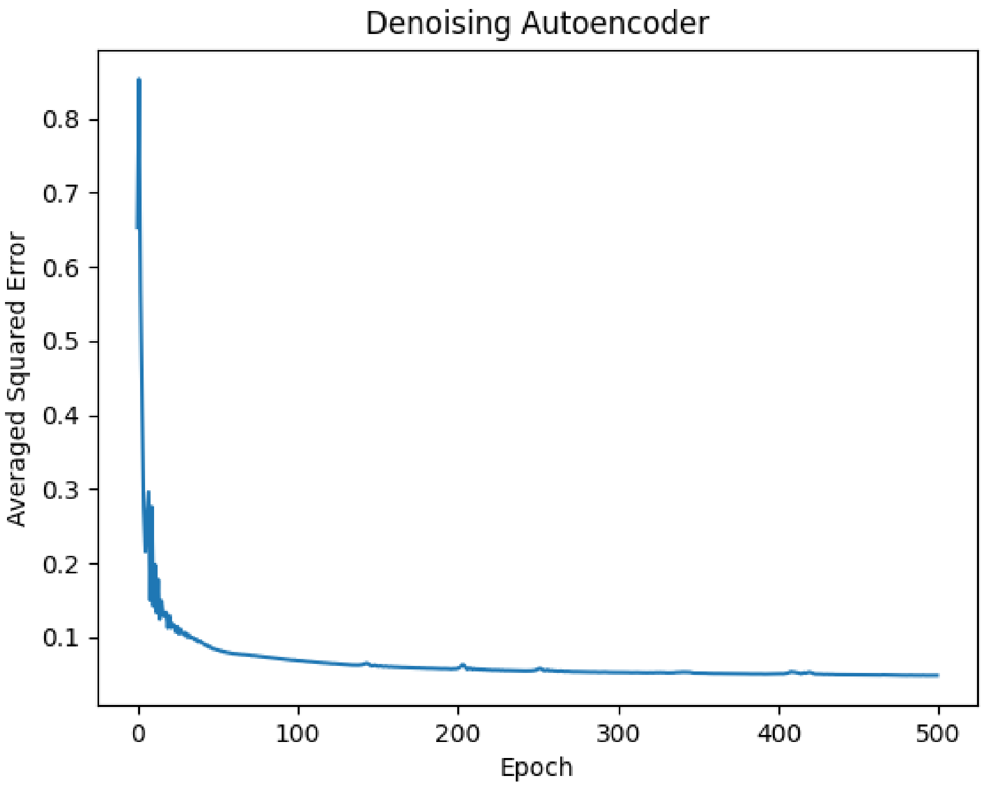
4880 neorons → 128 neorons → 488 neorons → 244 neorons → 488 neorons → 128 neorons → 4880 neorons

Mean squared error: 0.015185078433734 57

One can see that Architecture 2 has the best performance. The reason that Architecture 1 performs worse than Architecture 2 may be that it is not deep enough and therefore underfits the model. However, the deeper architecture doesn't imply the better performance. The reason that Architecture 3 performs worse than Architecture 2 may be that it is too deep and therefore overfits the model.

2.5 (e)

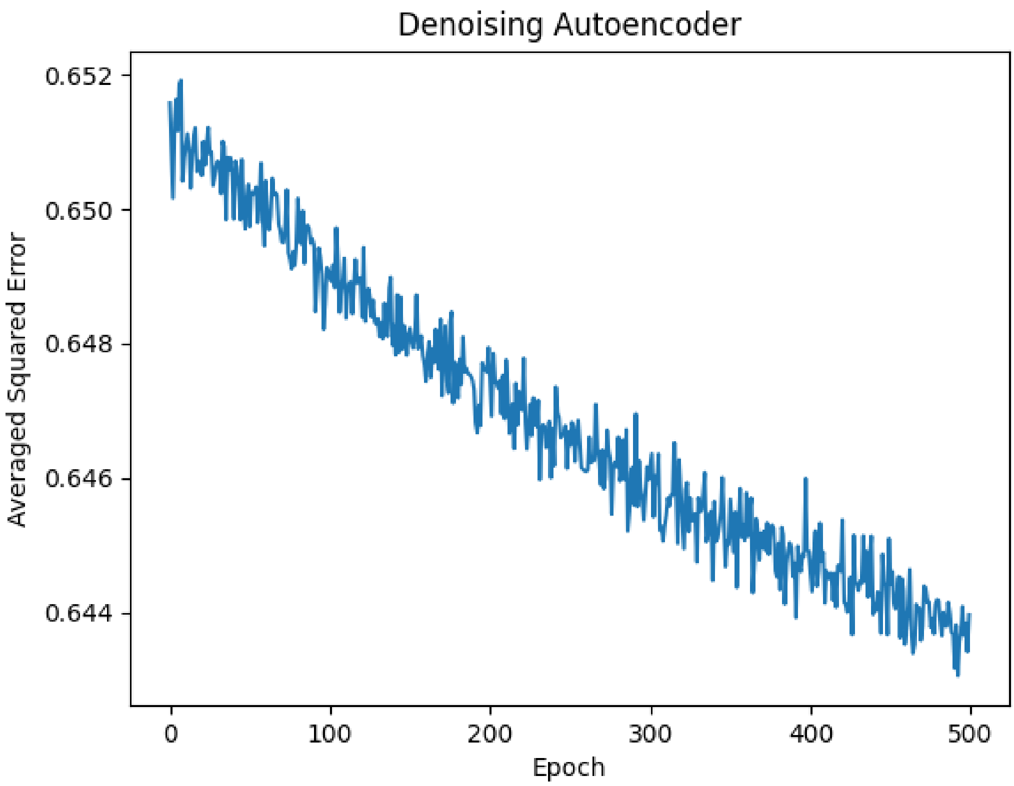
Adam:



Converged mean squared error: 0.013616964727733301.

Accurracy: 0.9333333333333333.

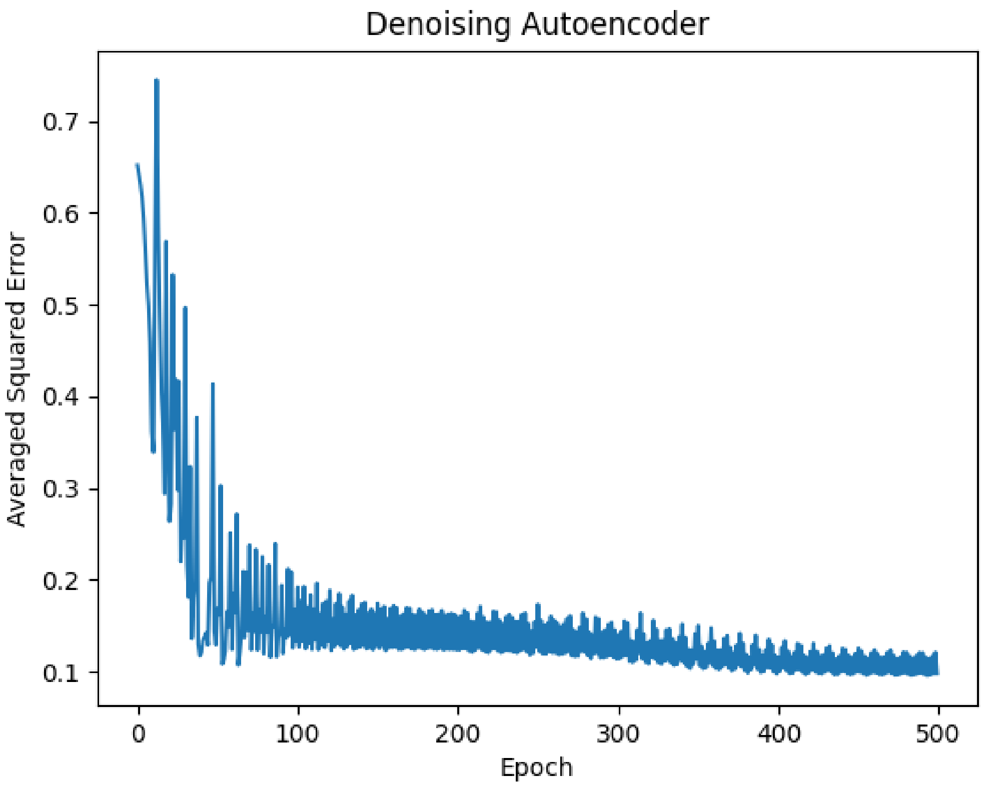
SGD:



Mean squared error: 0.6925974951675007.

Accurracy: 0.9.

Adadelta:



Converged mean squared error: 0.08281828092189876.

Accurracy: 0.7333333333333333.

One can see that the error of SGD decreases slowly, and it's mean squared error after 500 rounds is much more than the other two. Adam converges faster than the other two, and its performance (including error and accurracy) is also the best. Although SGD has high error, its accurracy is better than Adadelta. The reason may be that Adadelta overfits the model.