Machine Learning Fall 2020 ——— Homework 5

學號：B07902037 系級： 資工三 姓名：蔡沛勳

1. **(1%) 請使用不同的 Autoencoder model，以及不同的降維方式(降到不同維度)，討論其 reconstruction loss & public / private accuracy。（因此模型需要兩種，降維方法也需要兩種，但 clustrering 不用兩種。）**

   我用 CNN 和 DNN 實做了不同的 Autoencoder，具體結構如下：

   CNN:

```
class AE(nn.Module):
    def __init__(self):
        super(AE, self).__init__()

        self.encoder = nn.Sequential(
            nn.Conv2d(3, 64, 3, stride=1, padding=1),
            nn.ReLU(True),
            nn.MaxPool2d(2),
            nn.Tanh(),
            nn.Conv2d(64, 128, 3, stride=1, padding=1),
            nn.ReLU(True),
            nn.MaxPool2d(2),
            nn.Tanh(),
            nn.Conv2d(128, 256, 3, stride=1, padding=1),
            nn.ReLU(True),
            nn.MaxPool2d(2),
            nn.Tanh(),

            nn.Conv2d(256, 256, 3, stride=1, padding=1),
            nn.ReLU(True),
            nn.Tanh(),
            #nn.MaxPool2d(2)
            nn.Conv2d(256, 256, 3, stride=1, padding=1),
            nn.ReLU(True),
        )

        self.decoder = nn.Sequential(
            nn.ConvTranspose2d(256, 128, 5, stride=1),
            nn.ReLU(True),
            nn.Tanh(),
            nn.ConvTranspose2d(128, 64, 9, stride=1),
            nn.ReLU(True),
            nn.Tanh(),
            nn.ConvTranspose2d(64, 3, 17, stride=1),
            nn.Tanh()
        )

    def forward(self, x):
        x1 = self.encoder(x)
        x  = self.decoder(x1)
        return x1, x
```

DNN:

```
class AE2(nn.Module):
    def __init__(self):
        super(AE2, self).__init__()

        self.encoder = nn.Sequential(
            nn.Linear(3 * 32 * 32, 1024),
            nn.LeakyReLU(0.5),
            nn.Tanh(),
            nn.Linear(1024, 1024),
            nn.LeakyReLU(0.5),
            nn.Tanh(),
            nn.Linear(1024, 1024),
            nn.LeakyReLU(0.5),
        )

        self.decoder = nn.Sequential(
            nn.Linear(1024, 1024),
            nn.LeakyReLU(0.5),
            nn.Tanh(),
            nn.Linear(1024, 1024),
            nn.LeakyReLU(0.5),
            nn.Tanh(),
            nn.Linear(1024, 3 * 32 * 32),
            nn.LeakyReLU(0.5),
            nn.Tanh(),
        )

    def forward(self, x):
        x  = torch.reshape(x, (-1, 3 * 32 * 32))
        x1 = self.encoder(x)
        x  = self.decoder(x1)
        x  = torch.reshape(x, (-1, 3, 32, 32))
        return x1, x
```

兩 Autoencoder 皆以 Adam 在 learning rate = 0.00001 做 100 個 epoch。之後 CNN 得到結果為 4096 維的資料，DNN 結果為 1024 維的資料。兩者皆以 PCA 降到 128 維後再用 t-SNE 降到 2 維，最終以 K-Means 做分類。得到結果如下：

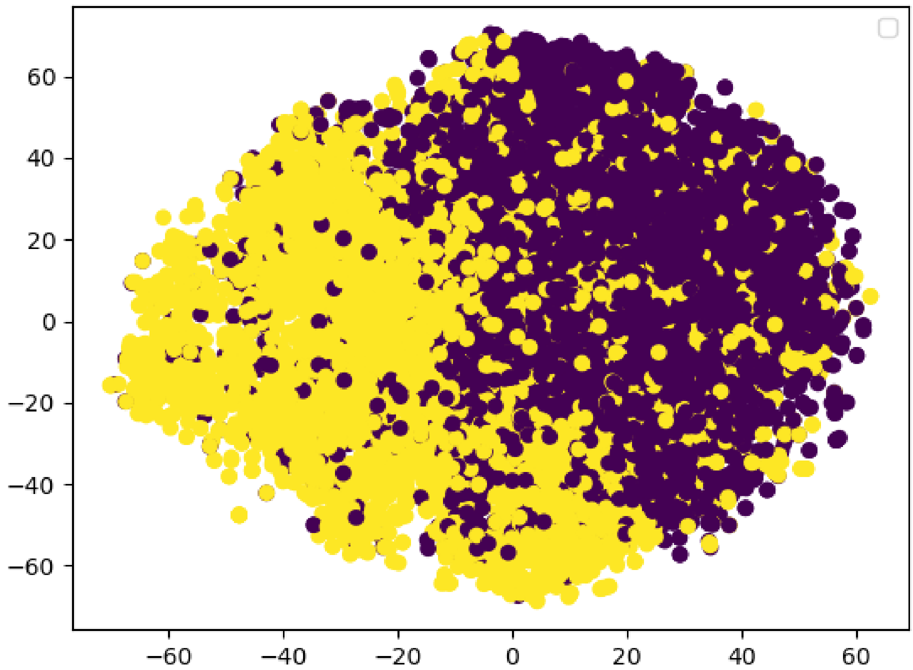
	reconstruction loss	accuracy
CNN model	7.41368	0.79589
DNN model	6.72101	0.81089

2. (1%) 從 dataset 選出 2 張圖，並貼上原圖以及經過 autoencoder 後 reconstruct 的圖片。

以下兩張圖片為 DNN 實作的 Autoencoder 原本即 reconstruct 後的結果比較，可以看出畫片中的內容再 reconstruct 後模糊不少。

3. (1%) 我們會給你 dataset 的 label。請在二維平面上視覺化 label 的分佈。

以下為 DNN 實作的 Autoencoder 再經過 PCA 及 t-SNE 降至二維後的分布情形。



4. (3%)Refer to math problem

1.

t = 1:

$$z = \left[0, 0, 0, 1\right] \cdot \left[0, 1, 0, 3\right] + 0 = 3$$
$$z_i = \left[100, 100, 0, 0\right] \cdot \left[0, 1, 0, 3\right] - 10 = 90$$
$$z_f = \left[-100, -100, 0, 0\right] \cdot \left[0, 1, 0, 3\right] + 110 = 10$$
$$z_o = \left[0, 0, 100, 0\right] \cdot \left[0, 1, 0, 3\right] - 10 = -10$$
$$c' = \frac{1}{1 + e^{-90}} \cdot 3 + 0 \cdot \frac{1}{1 + e^{-10}} \approx 3$$
$$y_1 = \frac{1}{1 + e^{10}} \cdot 3 \approx 0$$

t = 2:

$$z = \left[0, 0, 0, 1\right] \cdot \left[1, 0, 1, -2\right] + 0 = -2$$
$$z_i = \left[100, 100, 0, 0\right] \cdot \left[1, 0, 1, -2\right] - 10 = 90$$
$$z_f = \left[-100, -100, 0, 0\right] \cdot \left[1, 0, 1, -2\right] + 110 = 10$$
$$z_o = \left[0, 0, 100, 0\right] \cdot \left[1, 0, 1, -2\right] - 10 = 90$$
$$c' = \frac{1}{1 + e^{-90}} \cdot -2 + 3 \cdot \frac{1}{1 + e^{-10}} \approx 1$$
$$y_2 = \frac{1}{1 + e^{-90}} \cdot 1 \approx 1$$

t = 3:

$$z = \left[0, 0, 0, 1\right] \cdot \left[1, 1, 1, 4\right] + 0 = 4$$
$$z_i = \left[100, 100, 0, 0\right] \cdot \left[1, 1, 1, 4\right] - 10 = 190$$
$$z_f = \left[-100, -100, 0, 0\right] \cdot \left[1, 1, 1, 4\right] + 110 = -90$$
$$z_o = \left[0, 0, 100, 0\right] \cdot \left[1, 1, 1, 4\right] - 10 = 90$$
$$c' = \frac{1}{1 + e^{-190}} \cdot 4 + 1 \cdot \frac{1}{1 + e^{90}} \approx 4$$
$$y_3 = \frac{1}{1 + e^{-90}} \cdot 4 \approx 4$$

t = 4:

$$z = \left[0, 0, 0, 1\right] \cdot \left[0, 1, 1, 0\right] + 0 = 0$$
$$z_i = \left[100, 100, 0, 0\right] \cdot \left[0, 1, 1, 0\right] - 10 = 90$$
$$z_f = \left[-100, -100, 0, 0\right] \cdot \left[0, 1, 1, 0\right] + 110 = 10$$
$$z_o = \left[0, 0, 100, 0\right] \cdot \left[0, 1, 1, 0\right] - 10 = 90$$
$$c' = \frac{1}{1 + e^{-90}} \cdot 0 + 4 \cdot \frac{1}{1 + e^{-10}} \approx 4$$
$$y_4 = \frac{1}{1 + e^{-90}} \cdot 4 \approx 4$$

t = 5:

$$z = [0, 0, 0, 1] \cdot [0, 1, 0, 2] + 0 = 2$$
$$z_i = [100, 100, 0, 0] \cdot [0, 1, 0, 2] - 10 = 90$$
$$z_f = [-100, -100, 0, 0] \cdot [0, 1, 0, 2] + 110 = 10$$
$$z_o = [0, 0, 100, 0] \cdot [0, 1, 0, 2] - 10 = -10$$
$$c' = \frac{1}{1 + e^{-90}} \cdot 2 + 4 \cdot \frac{1}{1 + e^{-10}} \approx 6$$
$$y_5 = \frac{1}{1 + e^{10}} \cdot 6 \approx 0$$

t = 6:

$$z = [0, 0, 0, 1] \cdot [0, 0, 1, -4] + 0 = -4$$
$$z_i = [100, 100, 0, 0] \cdot [0, 0, 1, -4] - 10 = -10$$
$$z_f = [-100, -100, 0, 0] \cdot [0, 0, 1, -4] + 110 = 110$$
$$z_o = [0, 0, 100, 0] \cdot [0, 0, 1, -4] - 10 = 90$$
$$c' = \frac{1}{1 + e^{10}} \cdot -4 + 6 \cdot \frac{1}{1 + e^{-110}} \approx 6$$
$$y_6 = \frac{1}{1 + e^{-90}} \cdot 6 \approx 6$$

t = 7:

$$z = [0, 0, 0, 1] \cdot [1, 1, 1, 1] + 0 = 1$$
$$z_i = [100, 100, 0, 0] \cdot [1, 1, 1, 1] - 10 = 190$$
$$z_f = [-100, -100, 0, 0] \cdot [1, 1, 1, 1] + 110 = -90$$
$$z_o = [0, 0, 100, 0] \cdot [1, 1, 1, 1] - 10 = 90$$
$$c' = \frac{1}{1 + e^{-190}} \cdot 1 + 6 \cdot \frac{1}{1 + e^{90}} \approx 1$$
$$y_7 = \frac{1}{1 + e^{-90}} \cdot 1 \approx 1$$

t = 8:

$$z = [0, 0, 0, 1] \cdot [1, 0, 1, 2] + 0 = 2$$
$$z_i = [100, 100, 0, 0] \cdot [1, 0, 1, 2] - 10 = 90$$
$$z_f = [-100, -100, 0, 0] \cdot [1, 0, 1, 2] + 110 = 10$$
$$z_o = [0, 0, 100, 0] \cdot [1, 0, 1, 2] - 10 = 90$$
$$c' = \frac{1}{1 + e^{-90}} \cdot 2 + 1 \cdot \frac{1}{1 + e^{-10}} \approx 3$$
$$y_8 = \frac{1}{1 + e^{-90}} \cdot 3 \approx 3$$

2.

從題目得到

$$h = W^T x$$
$$u = W'^T x$$
$$y = Softmax(u) = Softmax\left(W'^T W^T x\right)$$
$$Loss = L = -\log \prod_{c \in C} P\left(w_{output,c}, w_{input}\right) = -\log \prod_{c \in C} \frac{\exp(u_c)}{\sum_{i \in V} \exp\,(u_i)}$$

設

$$W = \begin{bmatrix} W_{1,1} & W_{1,2} & \cdots & W_{1,N} \\ W_{2,1} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ W_{V,1} & \cdots & \cdots & W_{V,N} \end{bmatrix}$$

可推得

$$h = \begin{bmatrix} W_{1,1} & W_{2,1} & \cdots & W_{V,1} \\ W_{1,2} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ W_{1,N} & \cdots & \cdots & W_{V,N} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_V \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^{V} W_{k,1} x_k \\ \sum_{k=1}^{V} W_{k,2} x_k \\ \vdots \\ \sum_{k=1}^{V} W_{k,N} x_k \end{bmatrix}$$

$$u = \begin{bmatrix} W'_{1,1} & W'_{2,1} & \cdots & W'_{N,1} \\ W'_{1,2} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ W'_{1,V} & \cdots & \cdots & W'_{N,V} \end{bmatrix} \begin{bmatrix} \sum_{k=1}^{V} W_{k,1} x_k \\ \sum_{k=1}^{V} W_{k,2} x_k \\ \vdots \\ \sum_{k=1}^{V} W_{k,N} x_k \end{bmatrix} = \begin{bmatrix} \sum_{l=1}^{N} \left(W'_{l,1} \sum_{k=1}^{V} W_{k,l} x_k \right) \\ \sum_{l=1}^{N} \left(W'_{l,2} \sum_{k=1}^{V} W_{k,l} x_k \right) \\ \vdots \\ \sum_{l=1}^{N} \left(W'_{l,N} \sum_{k=1}^{V} W_{k,l} x_k \right) \end{bmatrix}$$

L 可轉換成

$$L = -\log \prod_{c \in C} \frac{\exp(u_c)}{\sum_{i \in V} \exp(u_i)} = -\sum_{c \in C} \log \frac{\exp(u_c)}{\sum_{i \in V} \exp(u_i)}$$
$$= -\sum_{c \in C} \left(\log\left(\exp(u_c)\right) - \log\left(\sum_{i \in V} \exp(u_i) \right) \right) = -\sum_{c \in C} \left(u_c - \log\left(\sum_{i \in V} \exp(u_i) \right) \right)$$
$$= -\sum_{c \in C} \left(\sum_{l=1}^{N} \left(W'_{l,c} \sum_{k=1}^{V} W_{k,l} x_k \right) - \log\left(\sum_{i \in V} \exp\left(\sum_{l=1}^{N} \left(W'_{l,i} \sum_{k=1}^{V} W_{k,l} x_k \right) \right) \right) \right)$$

L 對 $W_{i,j}^{T}$ 微分得

$$\frac{\partial L}{\partial W_{i,j}^{T}} = \frac{\partial L}{\partial W_{j,i}}$$

$$= -\frac{\partial}{\partial W_{j,i}} \sum_{c \in C} \left(\sum_{l=1}^{N} \left(W'_{l,c} \sum_{k=1}^{V} W_{k,l} x_k \right) - \log \left(\sum_{m \in V} \exp \left(\sum_{l=1}^{N} \left(W'_{l,m} \sum_{k=1}^{V} W_{k,l} x_k \right) \right) \right) \right)$$

$$= -\sum_{c \in C} \left(W'_{i,c} x_j - \frac{\sum_{m \in V} W'_{i,m} x_j \exp\left(\sum_{l=1}^{N} \left(W'_{l,m} \sum_{k=1}^{V} W_{k,l} x_k \right) \right)}{\sum_{m \in V} \exp\left(\sum_{l=1}^{N} \left(W'_{l,m} \sum_{k=1}^{V} W_{k,l} x_k \right) \right)} \right)$$

$$= -\sum_{c \in C} \left(W'_{i,c} x_j - \frac{\sum_{m \in V} W'_{i,m} x_j \exp(u_m)}{\sum_{m \in V} \exp(u_m)} \right)$$

$i \in C$ 的前提下， L 對 $W'^{T}_{i,j}$ 微分得

$$\frac{\partial L}{\partial W'^{T}_{i,j}} = \frac{\partial L}{\partial W'_{j,i}}$$

$$= -\frac{\partial}{\partial W'_{j,i}} \sum_{c \in C} \left(\sum_{l=1}^{N} \left(W'_{l,c} \sum_{k=1}^{V} W_{k,l} x_k \right) - \log \left(\sum_{m \in V} \exp \left(\sum_{l=1}^{N} \left(W'_{l,m} \sum_{k=1}^{V} W_{k,l} x_k \right) \right) \right) \right)$$

$$= -\left(\sum_{k=1}^{V} W_{k,j} x_k - \sum_{c \in C} \left(\frac{\left(\sum_{k=1}^{V} W_{k,l} x_k \right) \exp\left(\sum_{l=1}^{N} \left(W'_{l,m} \sum_{k=1}^{V} W_{k,l} x_k \right) \right)}{\sum_{m \in V} \exp\left(\sum_{l=1}^{N} \left(W'_{l,m} \sum_{k=1}^{V} W_{k,l} x_k \right) \right)} \right) \right)$$

$$= -\sum_{k=1}^{V} W_{k,j} x_k + \sum_{c \in C} \left(\frac{\left(\sum_{k=1}^{V} W_{k,j} x_k \right) \exp(u_i)}{\sum_{m \in V} \exp(u_m)} \right)$$