學號：R10921019　系級：電機所碩二　　姓名：鄭皓方

Kaggle Competetion

1. (1%) 請附上你在 kaggle 競賽上表現最好的降維以及分群方式，並條列**五種**不同降維維度的設定對應到的表現(public / private accuracy)

   註1: auto-encoder 和 PCA 只要任一維度不一樣即可算是一種組合。

   註2: 不限於以上方法，同學也可以使用任何其他 embedding algorithm 實現降維。

   我使用的autoencoder的架構如下，有三層結構對稱的convelution layer以及convelution transpose layer，夾在中間的是兩層linear 的layer，負責產生latent vector。產生的latent vector 會放進pca演算法再降至更低的維度，之後再用K-means 做clustering。

```
Net(
  (encoder): Sequential(
    (0): Conv2d(3, 8, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (1): ReLU()
    (2): Conv2d(8, 16, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (3): ReLU()
    (4): Conv2d(16, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (5): ReLU()
  )
  (fc1): Linear(in_features=512, out_features=32, bias=True)
  (fc2): Linear(in_features=32, out_features=512, bias=True)
  (decoder): Sequential(
    (0): ConvTranspose2d(32, 16, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), output_padding=(1, 1))
    (1): ReLU()
    (2): ConvTranspose2d(16, 8, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), output_padding=(1, 1))
    (3): ReLU()
    (4): ConvTranspose2d(8, 3, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), output_padding=(1, 1))
    (5): ReLU()
  )
        ))
```

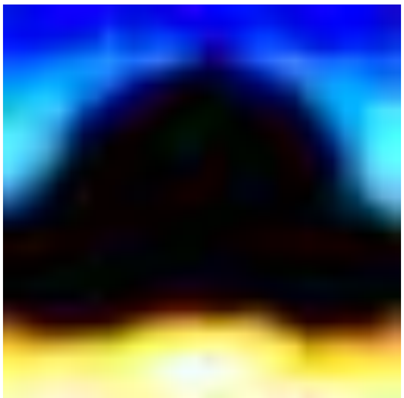
   下面我試著用不同的維度產生出降維的資料，我發現怎麼做調整好像都不會比原本的好，維度太高的話可能會造成下面一個分類器分類困難，維度太低的話可能又會損失一些重要的資訊。

Autoencoder 維度	PCA 維度	Private	Public
32	**8**	**0.81333**	**0.80666**
32	12	0.78889	0.78067

32	4	0.77888	0.76933
64	8	0.78223	0.77712
16	8	0.77022	0.764

2. (1%) 從 trainX.npy 選出不同類別的 2 張圖，貼上原圖以及你的 autoencoder reconstruct 的圖片。用 Mean Square Error 計算這兩張圖的 reconstruction error, 並說明該 error 與 kaggle score 的關係。

下面的第一組圖片主要是風景照的類別，產生的圖片跟原圖比較相似，error也比較小。第二組圖片是屬於另一種類別，reconstruction 的結果跟原圖不太一樣，error也比上一個model還高。在kaggle score的表現上，我的觀察是不一定rmse越小kaggle score就會越好。rmse越小代表latent vector 越能代表一張圖片。然而題目是要做分類器，如果兩種圖片都學到類似的特徵，在做分類的話就會比較困難。舉例來說，我在model加入batchnorm之後rmse有下降，但是kaggle score變差了，而且兩種圖片的rmse也較為接近。我認為最好的方式應該是第一組的rmse很低，然後跟第二組的差很多。

Oringinal	Reconstructed image
Group1 	mse err: 0.5353 

Group2	mse err: 0.7058
	

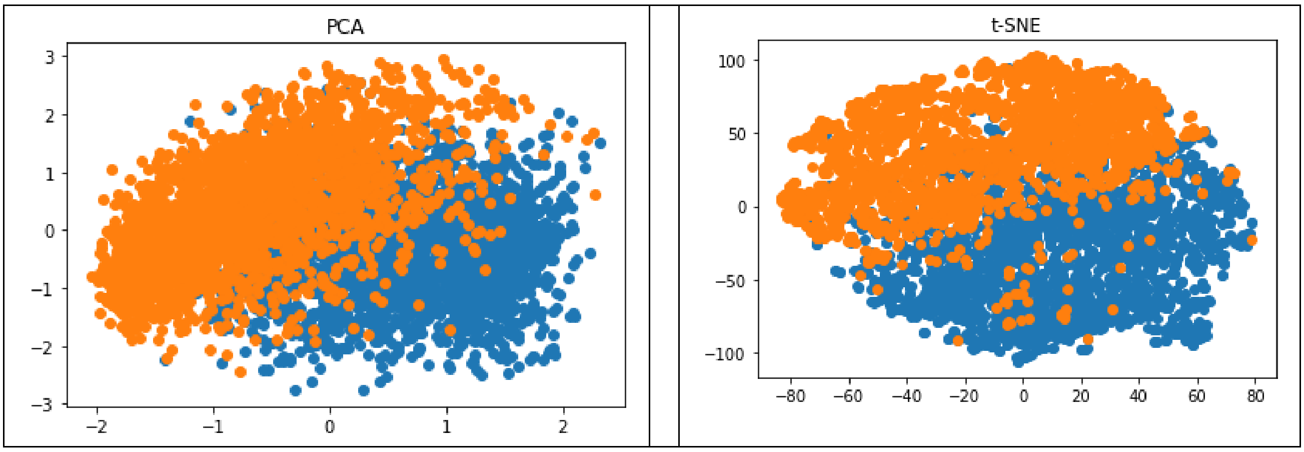
3. (2%) 請使用 pca 以及 tsne **兩種**方法, 將 visualization.npy 的圖片經過 autoencoder 降維後得到之 latent vector，進一步降維至二維平面並作圖。並說明兩張圖之差異。

註1: visualization.npy 前 2500 張 label 為 0；後 2500 張 label 為 1

註2: 一共要貼上2張圖片。

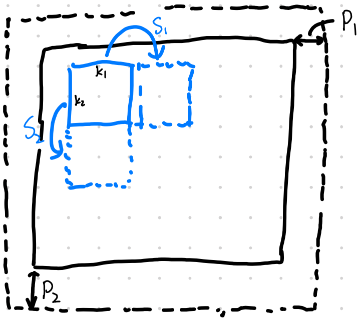
註3: 範例圖片如下（顏色、分佈不用完全一樣）

下圖為使用不同的方法將latent vector做分類所得到的結果，橘色是風景照的類別，可以發現不管用哪種方法都分佈的比較密集，可見他們具有比較共同的特徵，而這種特徵是可以被學習出來的。藍色的類別就比較分散，可見不是風景照的類別比較沒有共同的特徵。至於兩種降維方法的比較，我發現t-SNE對於風景照的分佈更為密集，而且看起來也分得比較散。推測可能的原因是t-SNE降維在低維度的時候會採用t分佈，比起高斯模型更注重長尾分佈，可以有效的將靠近中間的資料在映射後有較大的距離，可以稍微解決在降維後的擁擠問題。然而需要訓練的時間比起PCA差非常多，對於使用colab的我非常不友善，因此沒辦法在kaggle上有很好的實踐。



4. (4%) Refer to math problem：

1、convolution



(B, W, H, input_channels)

1. padding increase the both boundary
2. kernal size decrese on side boundary
3. stride would down sampling the image

$$H_{out} = \left\lfloor \frac{H + 2 \times P_1 - (k_1 - 1)}{S_1} + 1 \right\rfloor$$

$$W_{out} = \left\lfloor \frac{W + 2 \times P_2 - (k_2 - 1)}{S_2} + 1 \right\rfloor$$

2.
$$\frac{\partial l}{\partial \hat{x}_i} = \frac{\partial y_i}{\partial \hat{x}_i} \frac{\partial l}{\partial y_i} = \frac{\partial \gamma \hat{x}_i + \beta}{\partial \hat{x}_i} = \gamma \frac{\partial l}{\partial y_i}$$

$$\frac{\partial l}{\partial \sigma_\beta^2} = \sum_{i=1}^{m} \frac{\partial \hat{x}_i}{\partial \sigma_\beta^2} \frac{\partial l}{\partial \hat{x}_i} = \sum_{i=1}^{m} \frac{\partial \frac{x_i - \mu_\beta}{\sqrt{\sigma^2 + \varepsilon}}}{\partial \sigma^2} \gamma \frac{\partial l}{\partial y_i} = \sum_{i=1}^{m} -\frac{\gamma}{2} (x_i - \mu_\beta)(\sigma^2 + \varepsilon)^{-\frac{3}{2}} \frac{\partial l}{\partial y_i}$$

$$\frac{\partial l}{\partial \mu_\beta} = \sum_{i=1}^{m} \left(\frac{\partial l}{\partial \hat{x}_i} \frac{\partial \hat{x}_i}{\partial \mu_\beta} + \frac{\partial l}{\partial \sigma_\beta^2} \frac{\partial \sigma_\beta^2}{\partial \mu_\beta} \right) = \sum_{i=1}^{m} \left(\gamma \frac{\partial l}{\partial y_i} \cdot \frac{-1}{\sqrt{\sigma_\beta^2 + \varepsilon}} + \frac{\partial l}{\partial \sigma_\beta^2} \frac{-2(x_i - \mu_\beta)}{m} \right)$$

$$\frac{\partial l}{\partial x_i} = \frac{\partial l}{\partial \hat{x}_i} \frac{\partial \hat{x}_i}{\partial x_i} + \frac{\partial l}{\partial \mu_\beta} \frac{\partial \mu_\beta}{\partial x_i} + \frac{\partial l}{\partial \sigma_\beta^2} \frac{\partial \sigma_\beta^2}{\partial x_i}$$

$$= \frac{\partial l}{\partial \hat{x}_i} \cdot \frac{1}{\sqrt{\sigma_\beta^2 + \varepsilon}} + \frac{\partial l}{\partial \mu_\beta} \cdot \frac{1}{m} + \frac{\partial l}{\partial \sigma_\beta^2} \cdot \frac{2(x_i - \mu_\beta)}{m}$$

$$\frac{\partial l}{\partial \beta} = \sum_{i=1}^{m} \frac{\partial y_i}{\partial \beta} \frac{\partial l}{\partial y_i} = \sum_{i=1}^{m} \frac{\partial l}{\partial y_i}$$

$$\frac{\partial l}{\partial \gamma} = \sum_{i=1}^{m} \frac{\partial l}{\partial y_i} \frac{\partial y_i}{\partial \gamma} = \sum_{i=1}^{m} \hat{x}_i \cdot \frac{\partial l}{\partial y_i}$$

3、softmax

1、 $S_j = \frac{e^{z_j}}{\sum_{k=1}^{N} e^{z_k}} \Rightarrow \frac{\partial S_i}{\partial z_j} = \frac{\partial \frac{e^{z_i}}{\sum_{k=1}^{N} e^{z_i}}}{\partial z_j}$

if $i \neq j$ $e^{z_i} = \text{const}$ $\frac{\partial S_i}{\partial z_j} = \frac{-e^{z_i} e^{z_j}}{(\sum_{k=1}^{N} e^{z_i})^2} = -S_i S_j$

if $i = j$ $e^{z_i} = e^{z_j}$ $\frac{\partial S_i}{\partial z_j} = S_i - \frac{e^{z_i} e^{z_j}}{(\sum_{k=1}^{N} e^{z_i})^2} = (1-S_j) S_i$

import $\delta_{ij} \Rightarrow \frac{\partial S_i}{\partial z_j} = (\delta_{ij} - S_j) S_i$ #

2. $L(y, \hat{y}) = -\sum_i y_i \log(\hat{y}_i)$

$$\frac{\partial L}{\partial z_i} = -\sum y_i \frac{\partial \log(S_i)}{\partial z_i} = -\sum y_i \frac{(1-S_i) S_i}{S_i} = -\sum y_i - S_i$$

$$= -\sum (y_i - \hat{y}_i)$$ #

4. 1. let $\mu = 0$

than $\frac{1}{n}\sum_{i=1}^{n}(x_i)(x_i)^T = \Sigma = U\Lambda U^T$

$\Lambda = diag(\lambda_1, \lambda_2 \cdots \lambda_n)$, and Λ is a symmetric matrix

$\Rightarrow \Lambda^T = \Lambda$

$\Sigma = U\Lambda^{\frac{1}{2}}\Lambda^{\frac{1}{2}}U^T = (U\Lambda^{\frac{1}{2}})(\Lambda^{\frac{1}{2}T}U^T)$ where $\Lambda^{\frac{1}{2}} = diag(\pm\sqrt{\lambda_1}, \pm\sqrt{\lambda_2} \cdots \pm\sqrt{\lambda_n})$,

and $\lambda_1 \le \lambda_2 \le \lambda_3 \le \lambda_4 \cdots \le \lambda_m$

$x_i = \sqrt{n}U\Lambda^{\frac{1}{2}}$ $\quad X_i = \sqrt{n}U\Lambda^{\frac{1}{2}}$ ※

2. $Tr(\Phi\Sigma\Phi) = \frac{1}{N}\|\Phi^T U\Lambda^{\frac{1}{2}}\|_F^2$

let $\Phi = [u_1, u_2, u_3 \cdots u_k] \in R^{m\times k}$

$$Tr(\Phi\Sigma\Phi) = \left\| \overset{k\times m}{\begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_k \end{bmatrix}} \overset{m\times m}{[u_1, u_2 \cdots u_m]} \Lambda^{\frac{1}{2}} \right\|_2^F$$

$$= \left\| [\overset{k\times k}{I_k} | \overset{k\times m-k}{0}] \begin{bmatrix} \pm\sqrt{\lambda_1} & & & \\ & \pm\sqrt{\lambda_2} & & \\ & & \ddots & \\ & & & \pm\sqrt{\lambda_m} \end{bmatrix} \right\|_2^F$$

$= \sum_{i=1}^{k}\lambda_i$, where $\lambda_1 < \lambda_2 < \lambda_2 \cdots < \lambda_m$

this result give the smallest set of eigen value, therefore, it's a minimum solution