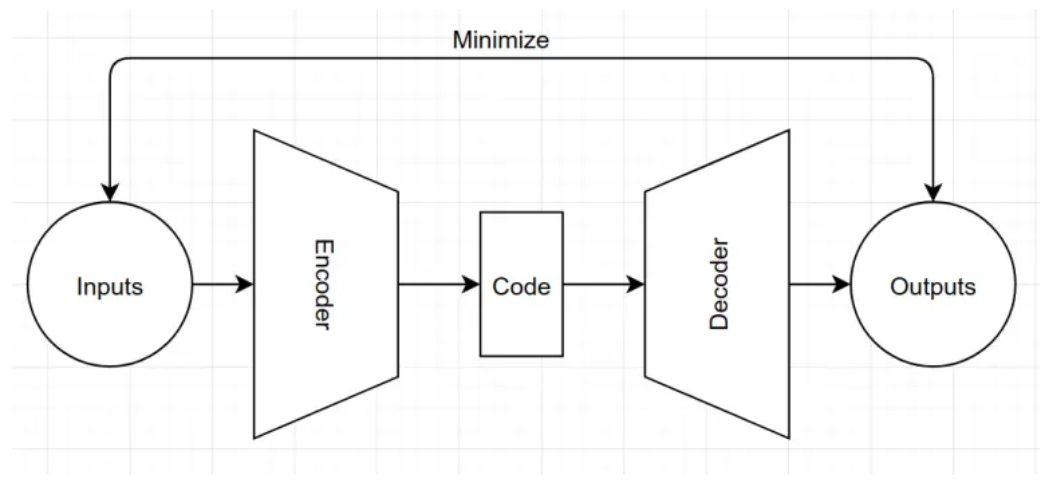
HW2 Report\_ Convolutional Autoencoder\_109064518\_高聖哲

1. Model Arichitecture

Gaussian Noise

-data\_npy格式 :(1281,26,26,3) → gen\_data\_npy格式 :(105,61,26,26,3)

-label\_npy格式:(1281,1) → gen\_label\_npy格式 :(105,61,1)

1. Model敘述:
   1. 架構中可細分為 Encoder（編碼器）和 Decoder（解碼器）兩部分，它們分別做壓縮與解壓縮的動作，讓輸出值和輸入值表示相同.
   2. 在AutoEncoder過程中增加了一些限制，使生成向量遵從高斯分佈.由於高斯分佈可以通過其mean 和 standard deviation 進行參數化，因此是可以讓我們控制要生成的圖片。
2. Model building block

input\_size(int) –輸入訊號的通道數

output\_size(int) –卷積產生的通道數

kerner\_size(int or tuple) – 卷積核的大小

stride(int or tuple,optional) – 卷積步長，即要輸入擴大的倍數

padding(int or tuple, optional) –輸入的每一條邊補充0的層數,高寬都增加2\*padding

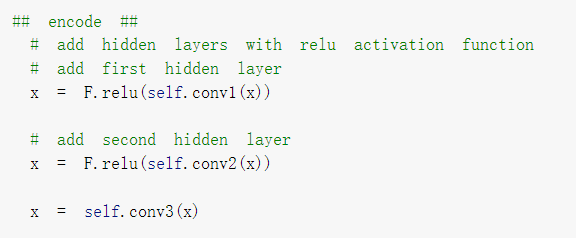
outpadding(int or tuple, optional) – 輸出邊補充0的層數，高寬都增加padding

* 1. **Encoder**
     1. **Conv2D公式:** (inputsize-kernel+2\*padding)/stride+1

-conv layer:(depth from 3 --> 16), 3x3 kernels

-conv layer:(depth from 16 --> 32), 3x3 kernels

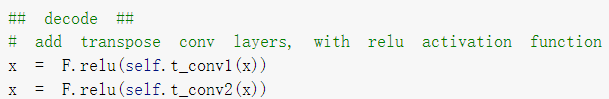
-conv layer:(depth from 32 --> 64), 5x5 kernels

* + 1. **透過Relu 的activation function 來新增hidden layer**
  1. **Decoder**
     1. **ConvTranspose2d公式:**(inputsize-1)\*stride+kernel-2\*padding+outpadding

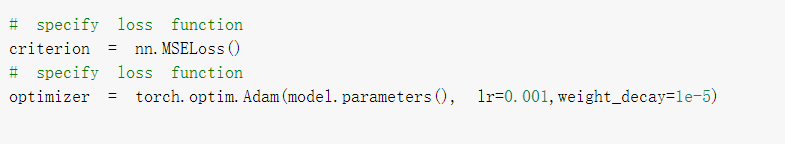
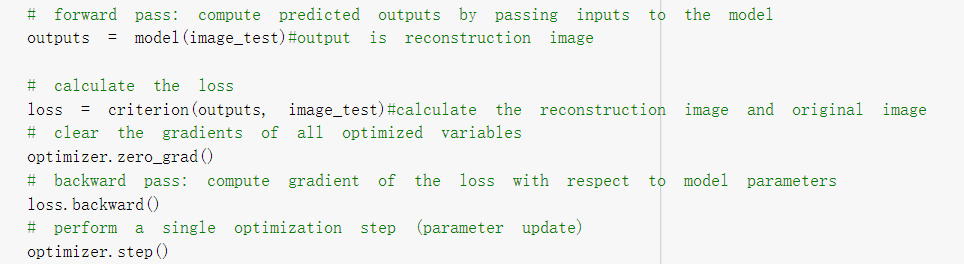
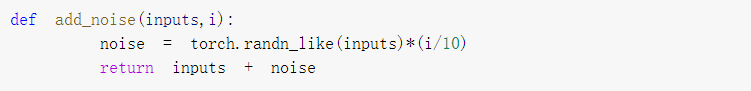
-tconv layer (depth from 64 --> 32), 5x5 kernels

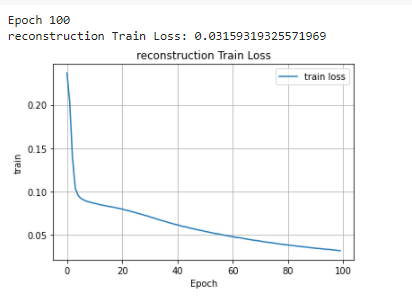
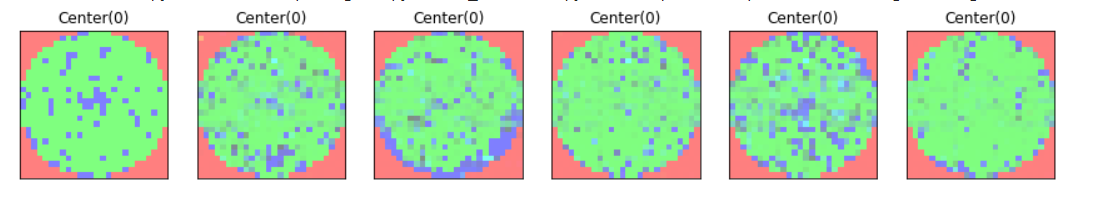
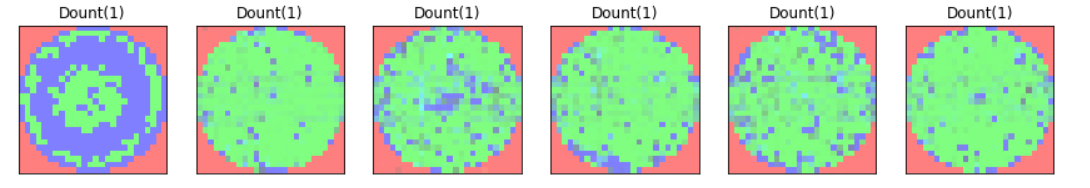
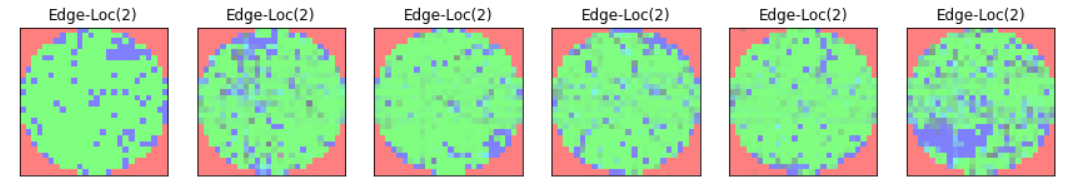
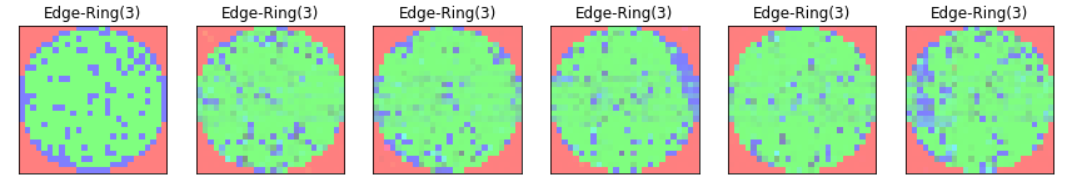
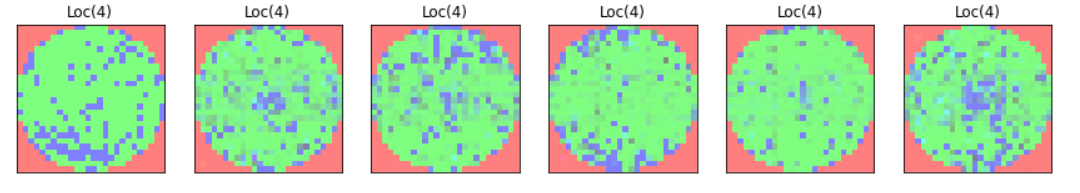
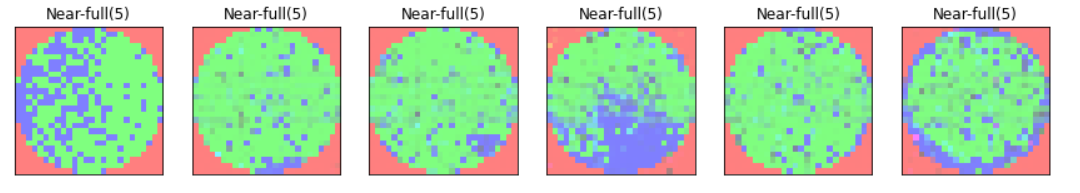
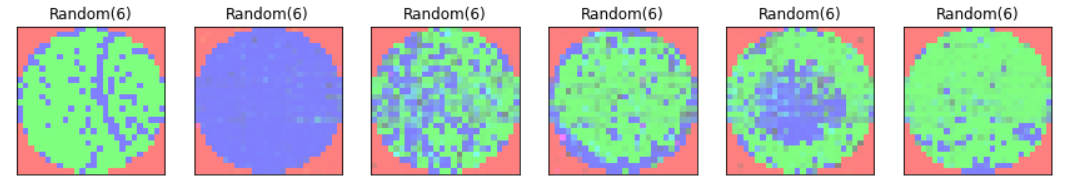
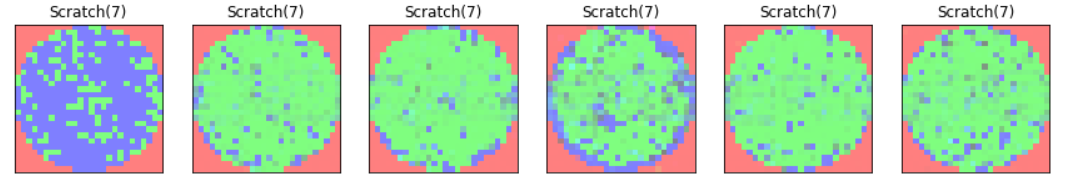
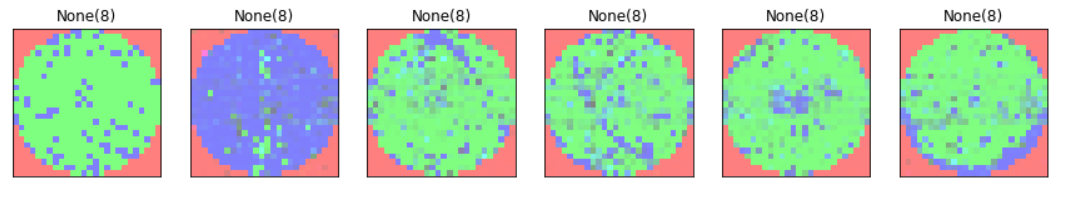
-tconv layer (depth from 32 --> 16), 3x3 kernels

-tconv layer (depth from 16 --> 3), 3x3 kernels

* + 1. **透過Relu 的activation function 來新增transpose con layer**
    2. **使用sigmoid作為output layer,最後的x輸出剛好等於input的size,則可以做後續的loss計算**
  1. **Loss function**
     1. nn.MSELoss均方損失函数：

,這裡的loss,x,y的維度是一樣的,可以是向量或是矩陣，ｉ是下標.比如若x,y,是矩陣

* + 1. Adam演算法：利用梯度的一階矩估計和二階矩估計動態調整每個引數的學習
    2. Output=model(new\_image1):對應前向傳播求出預測值
    3. loss = criterion(outputs, new\_image1):求當前的loss值
    4. optimizer.zero\_grad():梯度置零，也就是把loss關於weight的導數變成0.
    5. loss.backward():對應反向傳播求梯度
    6. optimizer.step():對應更新所有參數
  1. **Gaussian noise**
     1. **torch.randn\_like()函式:創建和input同樣尺寸的noisy tensor來達到Gaussian 的效果,並加add\_noise function加到autoencoder的latent code中**

1. Model模型結果
   1. Learning rate:0.001
   2. Weight-decay:1e-5
   3. Epoach:100
   4. Batchsize:61
   5. Reconstruction training loss: 0.03159319325571969
   6. Training figure:
2. Visualize 5 generated samples for each class
   1. **Center(0)**
   2. **Dount(1)**
   3. **Edge-Loc(2)**
   4. **Edge-Ring(3)**
   5. **Loc(4)**
   6. **Near-full(5)**
   7. **Random(6)**
   8. **Scratch(7)**
   9. **None(8)**