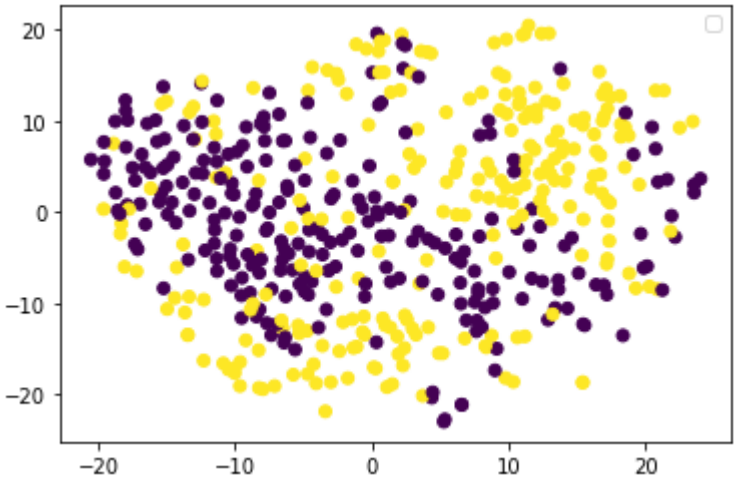
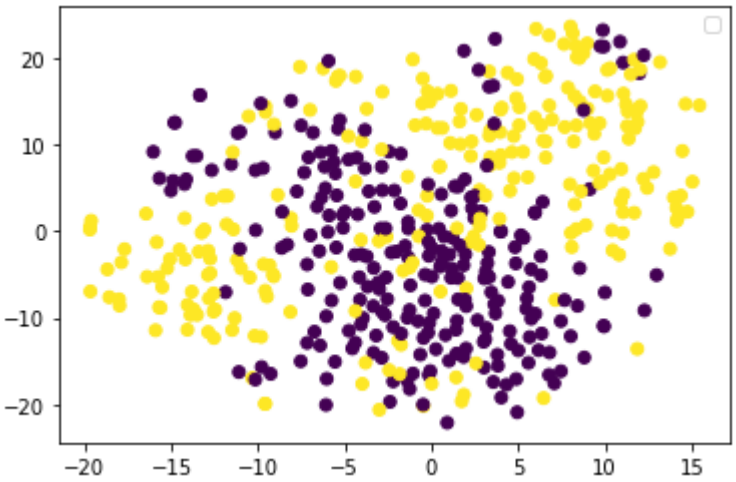
學號：b07901112　　系級：電機二　　姓名：劉聿珉

1. (3%) 請至少使用兩種方法 (autoencoder 架構、optimizer、data preprocessing、後續降維方法、clustering 算法等等) 來改進 baseline code 的 accuracy。
   a. 分別記錄改進前、後的 test accuracy 為多少。
      Baseline code 的 test accuracy = 0.74095
      Imporved 的 test accuracy = 0.80341(改進 optimizer)
      Best 的 test accuracy = 0.85200(改進 optimizer + autoencoder 架構)

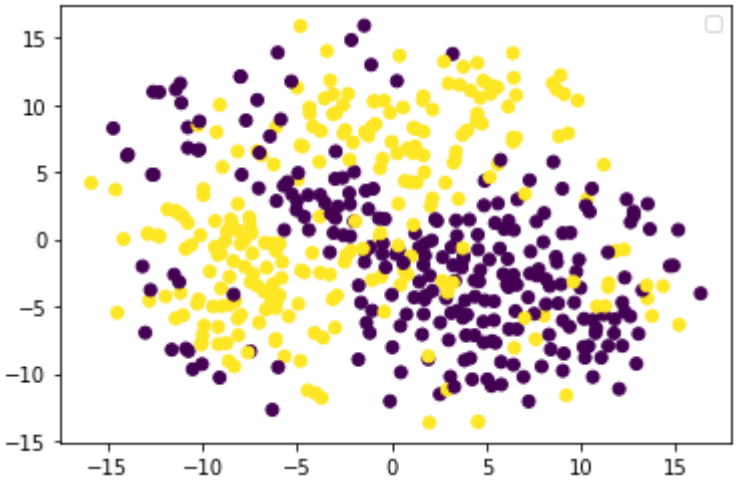
   b. 分別使用改進前、後的方法，將 **val data** 的降維結果 (embedding) 與他們對應的 label 畫出來。
      Baseline code：



      Imporved



Best



這幾張圖似乎是因為 random 的問題所以每次做出來都有點不一樣，但是可以看出紫色的點點跟黃色的點點有隨著正確率的提升重疊的部分也漸漸變少。

c. 盡量詳細說明你做了哪些改進。

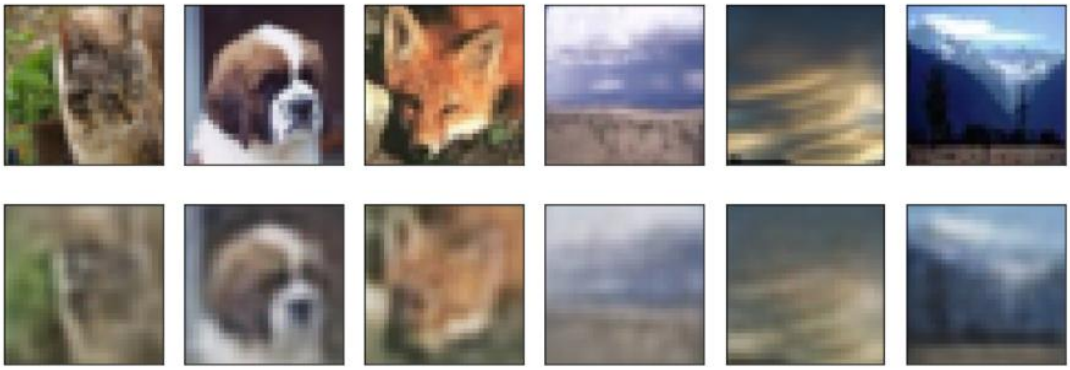
```
self.encoder = nn.Sequential(
    nn.Conv2d(3, 64, 3, stride=1, padding=1),
    nn.BatchNorm2d(64),
    nn.ReLU(True),
    nn.Dropout(0.2),
    # nn.MaxPool2d(2),
    nn.Conv2d(64, 128, 3, stride=1, padding=1),
    nn.BatchNorm2d(128),
    nn.ReLU(True),
    nn.Dropout(0.2),
    # nn.MaxPool2d(2),
    nn.Conv2d(128, 256, 3, stride=1, padding=1),
    nn.BatchNorm2d(256),
    nn.ReLU(True),
    nn.Dropout(0.2),
    nn.MaxPool2d(2),
    nn.Conv2d(256, 512, 3, stride=1, padding=1),
    nn.BatchNorm2d(512),
    nn.ReLU(True),
    nn.Dropout(0.2),
    nn.MaxPool2d(2),
    nn.Conv2d(512, 512, 3, stride=1, padding=1),
    nn.BatchNorm2d(512),
    nn.ReLU(True),
    nn.Dropout(0.2),
    nn.MaxPool2d(2)
)

self.decoder = nn.Sequential(
    nn.ConvTranspose2d(512, 256, 5, stride=1),
    nn.ReLU(True),
    nn.ConvTranspose2d(256, 128, 9, stride=1),
    nn.ReLU(True),
    nn.ConvTranspose2d(128, 64, 9, stride=1),
    nn.ReLU(True),
    nn.ConvTranspose2d(64, 3, 9, stride=1),
    nn.Tanh()
)
```

第一個改良我只有改變 optimizer 以及將 epoch 加大變成 150，在一開始時我將 learning rate 設為 1e-4，隨著 epoch 增加，漸漸將 learning rate 減小，最終撿到 1e-6，並且在最後 25 個 epoch 將 optimizer 改成 SGD，momentum=0.9。因為看網路上有人說好像 SGD 雖然收斂的速度比較慢，但是比起 Adam 更可以到達最低點，所以我就再最後 25 個 epoch 用 SGD 做最後衝刺。

第二個改進(我的 best)則是建立在第一個改良的基礎上，將 autoencoder 架構改變，結構如左圖，我將 encoder 改成 5 層並且加入 dropout, BatchNorm2d。而 decoder 則改成四層。以上就是我的所有改進。

2. (1%) 使用你 test accuracy 最高的 autoencoder，從 trainX 中，取出 index 1, 2, 3, 6, 7, 9 這 6 張圖片

a. 畫出他們的原圖以及 reconstruct 之後的圖片。

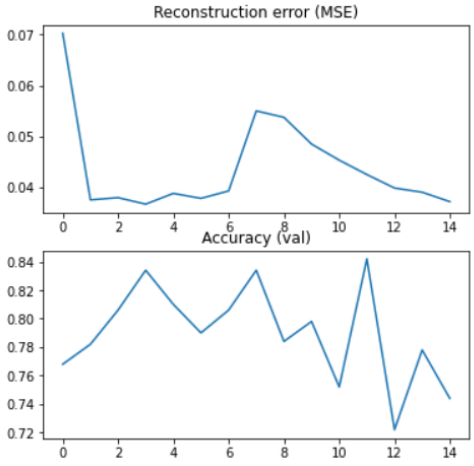


我們可以看到上下圖其實還原的程度算滿高的，小狗跟狐狸基本上可以輕易的辨認出來，而其他的四張圖則可以看出大部分的輪廓。

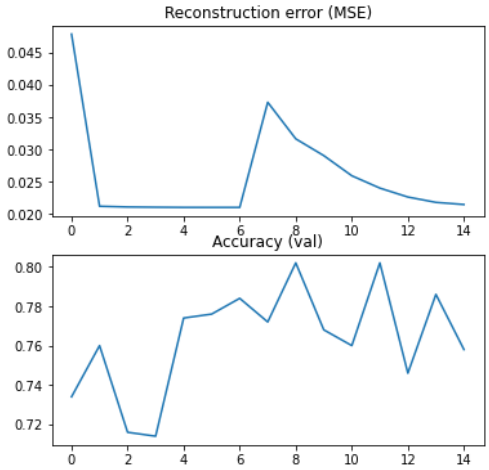
3. (2%) 在 autoencoder 的訓練過程中，至少挑選 10 個 checkpoints

a. 請用 model 的 train reconstruction error (用所有的 trainX 計算 MSE) 和 **val accuracy** 對那些 checkpoints 作圖。

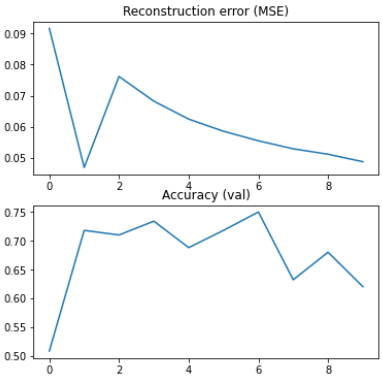
Improved 的作圖



Best 的作圖



Baseline 的作圖



b. 簡單說明你觀察到的現象。

MSE 的那幾張圖因為 checkpoints 編號的關係，所以 10 以後的 checkpoint 都會被平移到編號 0 之前(imporved 跟 Best 因為是 train150 個 epoch 所以有五個 check point 需要移到最後面(1~6)，而 Baseline 因為只有 train100 個 epoch，所以只有一個 checkpoint 需要搬(1))，所以看起來曲線一開始會有一個陡降，之後又會有一個險升。但如果將那些 checkpoint 平移到圖形的最後，做成正確的圖，我們可以發現它是一個隨著 X 軸增加 Y 軸漸減緩降坡(這部分真的非常抱歉，我時間有點趕，數據都是正確的只是需要平移一下)，這也表示作業二中的圖越接近原本的照片。

而 Acc 的部分跟 MSE 的問題一樣，需要做一點平移才會是對的圖。圖形在平移之後呈現一個鋸齒狀，我發現在我切換 optimizer 的時候 Acc 都會降低許多我在整個訓練中切換了四次，所以的鋸齒狀況非常明顯。(這張圖的圖形都呈現嚴重的鋸齒狀，有點難做細節的討論，但是依舊可以看出 acc 有比 train 之前增加許多)