DLCV HW4 Report

姓名：陳泓均
Collaborators：潘彥銘、林奕廷、詹書愷

Problem 1. Trimmed action recognition w/o RNN

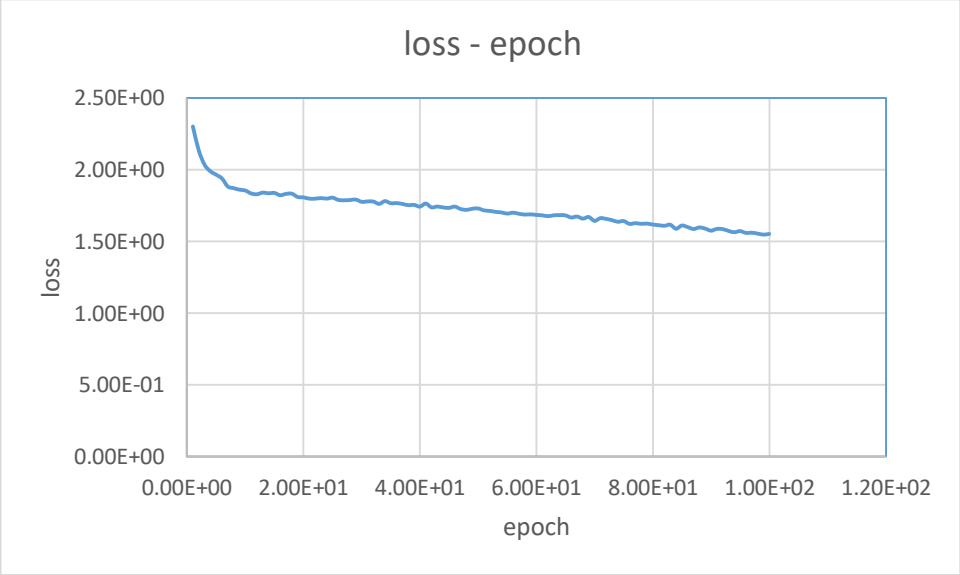
1. strategies of extracting CNN-based video features, training the model and other implementation details
   - extraction strategy 方面，我是抽取第一張以及最後一張的 feature 來做。使用的 feature 則是 resnet50 pretrain 在 ImageNet 上面的 feature。也因此在抽取 feature 之前我會先把圖片 resize 到 224x224。
   - Model structure 方面，基本上就是分為 feature extractor 和 classifier 兩部分，而 feature extractor 就是抽取 resnet pretrained feature。classifier 則是三層簡單的 fully connected layer，並加上 dropout 層避免 overfitting。

```
class label_cl(nn.Module):
    def __init__(self, num_classes):
        super(label_cl, self).__init__()
        self.hidden1 = nn.Linear(2048*2, 512)
        self.hidden2 = nn.Linear(512, 512)
        self.hidden3 = nn.Linear(512, num_classes)
        # self.batch = nn.BatchNorm1d(512)
        self.dropout1 = nn.Dropout(0.4)
        self.dropout2 = nn.Dropout(0.4)

    def forward(self, x):
        x = F.relu(self.dropout1(self.hidden1(x)))
        x = F.relu(self.dropout2(self.hidden2(x)))
        x = self.hidden3(x)

        return x
```

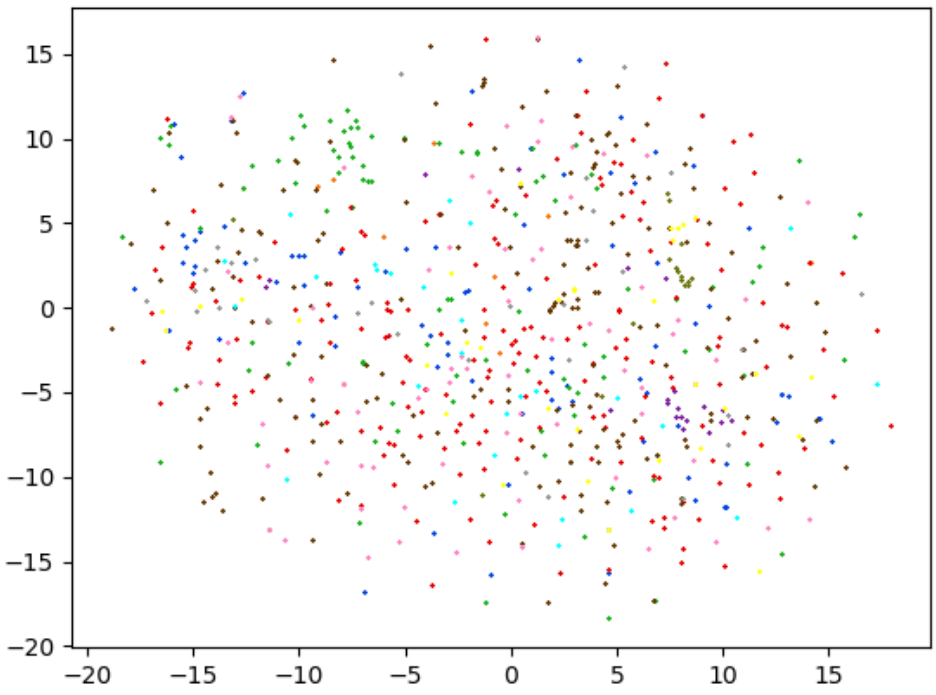
   - Model training 方面，我是訓練 100 個 epoch，使用 SGD optimizer，lr 為 0.002，分別在第 2、6、10 個 epoch 時會下降 0.32 倍。
   - Training set loss curve:



2. Validation performace

```
acc = 0.33940182054616386
```

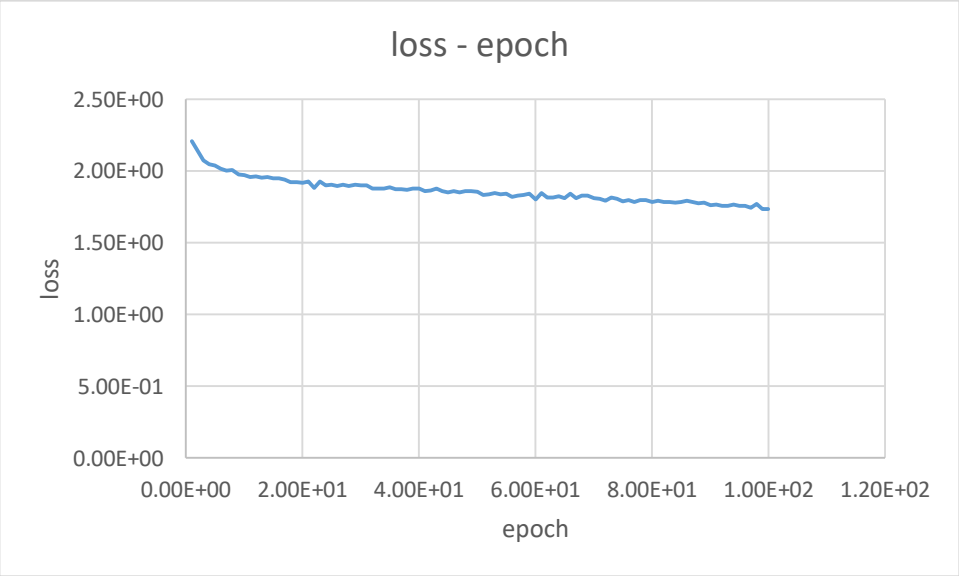
3. T-SNE visualization



從上圖可以看出，其實單純抽取 CNN 的 feature，並不會使得不同類別的 feature 分得很開，反而分布的滿分散的，因此 classification performance 並不會非常好。

Problem 2. Trimmed action recognition w/ RNN

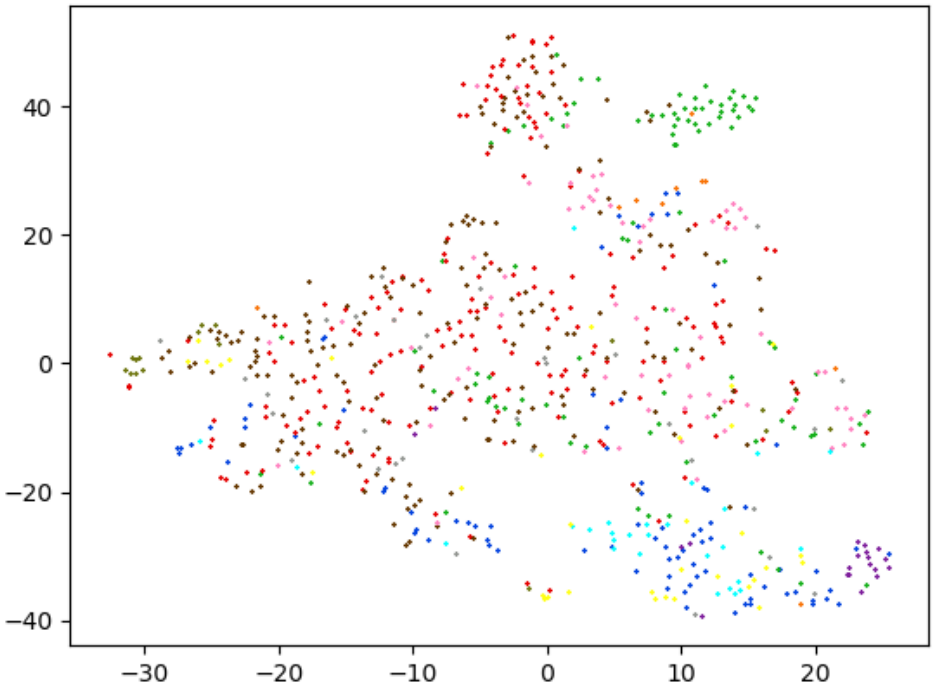
1. Describe your RNN models and implementation details for action recognition and plot the learning curve of your model:
   - RNN model: 我使用的 RNN model 是雙層的雙向 LSTM，hidden_dim = 512，使用最後一個 cell state 作為最後的 feature，並且把雙向的 output concat 起來，也因此 output 到 classifier 的 dimension 是 2 倍的 hidden_dim，也就是 1024。
   - Classifier: Classifier 的部分，直接使用一層 fully connected layer，因為我認為既然使用了 RNN，重點應該在把 RNN 訓練得更好，因此分類器的結構能夠輕量化一點。
   - Feature: 抽取的 feature 就和第一題一樣，且我有試過使用 resnet 101 或者 152，但 performance 似乎沒有顯著的提升。
   - Training: 訓練的方面，我使用的是 Adam Optimizer，learning rate = 0.001，且在 2,6,12,18 epoch 時下降 0.3 倍。
   - Training set Loss curve:



2. Validation performance

```
acc = 0.4668400520156047
```

3. T-SNE visualization



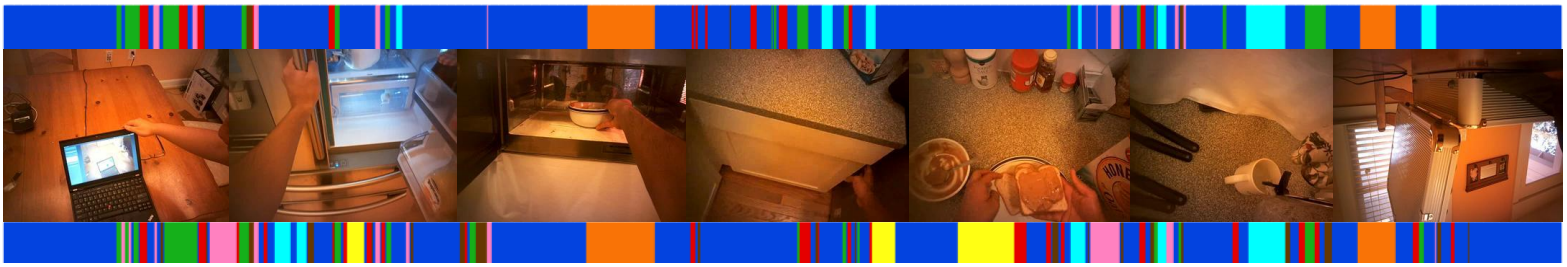
由上圖可以看到，RNN 的 feature 比起單純 CNN 的 feature 稍微進步了一點，例如可以看到右上角一塊綠色的區域是分得不錯，雖然有些類別例如紅色還是很分散，但比起 CNN 至少稍微有分類到，不會完全散在各處。不過也因為在 latent space 還是沒有分得很開，也因此準確率也還是不夠高。
而我認為之所以 RNN feature 比 CNN 好的原因在於它有時序的資訊，CNN feature 只是單純取幾個 frame，machine 比較難自己學到他們的關係，但 RNN 可以直接地告訴 machine 時序的資訊，並且一次 predict 的過程也觀察到比較多 frame，因此很理所當然可以預期有比較好的結果。

Problem 3. Temporal action segmentation

1. Describe any extension of your RNN models, training tricks, and post-processing techniques:
   基本上我沒有做任何改變，直接把第二題的 RNN model 和 classifier 套用到這題，有做改變的只有 loss function 而已；甚至 training 的參數都沒有做調整。至於 post-processing 我也完全沒有使用，但我觀察 predict 的結果發現其實她自己有某種程度上學到前後的值基本上會一樣，很少會有單獨出現的數值。
2. Validation accuracy
   ```
   5489/8938(0.6141194898187514)
   ```
   - ```
     503/1012(0.49703557312252966)
     ```
     (OP01-R02)
     ```

- 562/982(0.5723014256619144) (OP01-R04)
- 1624/2471(0.6572237960339944) (OP01-R07)
- 534/889(0.6006749156355455) (OP03-R04)
- 729/1085(0.6718894009216589) (OP04-R04)
- 568/948(0.5991561181434599) (OP05-R04)
- 969/1551(0.6247582205029013) (OP06-R03)

3. Visualization Results:



此為 OP04-R04 整部影片的 visualization result，validation accuracy 有 67%。由於我 training 的時候是一次拿 256 個 frame 去做 predict，我 testing 的時候也是用一樣的方式，把影片切成一段一段做 prediction。