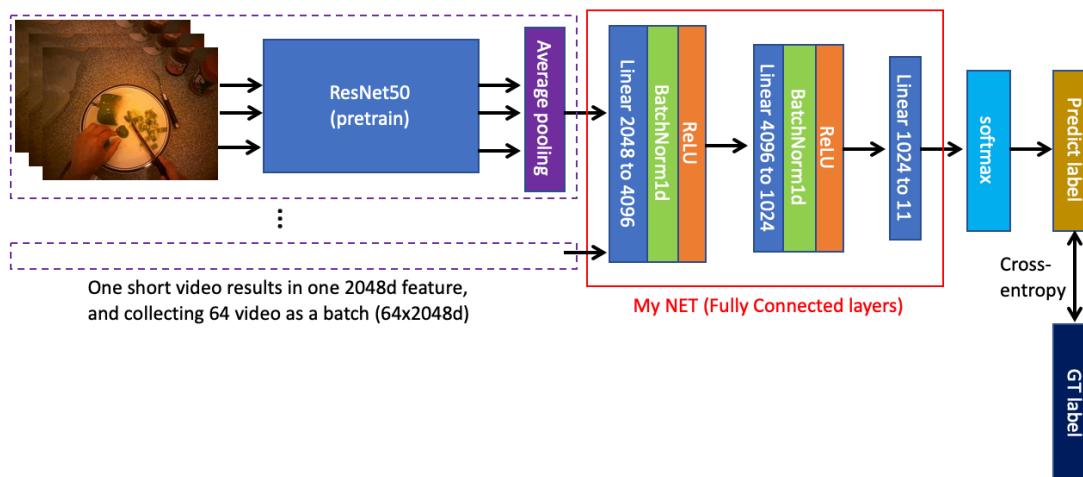


DLCV hw4 r07921001 李尚倫

Problem 1 : Data preprocessing (20%)

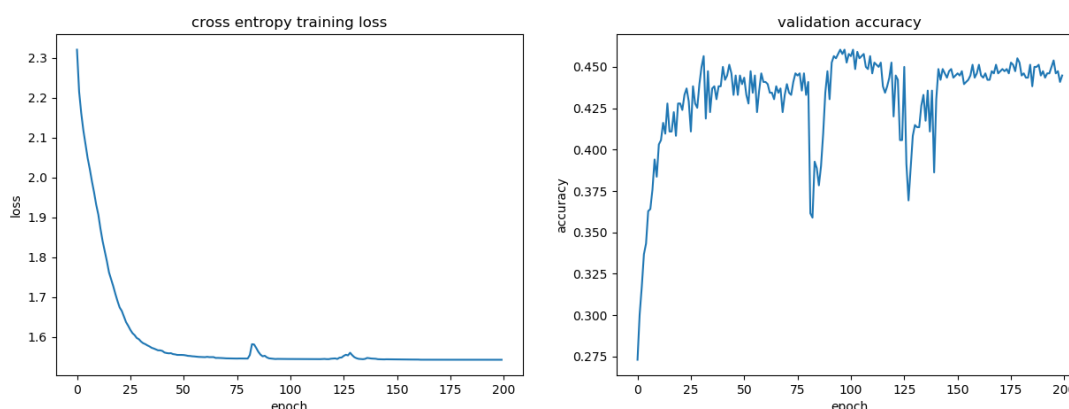
- Describe your strategies of extracting CNN-based video features, training the model and other implementation details (which pretrained model) and plot your learning curve (The loss curve of training set is needed, others are optional). (5%)



Discussion:

這題的training 主要分成兩個步驟，第一個步驟是用pretrain 在ImageNet上的model ResNet50來對p1的所有小短片的每個frame抽出2048d的feature，然後做average pooling，每支短片得到一個2048d的feature依序存在list中，最後轉成torch並存成.pt檔。

第二步驟則是再將這些feature load近來，並以一個batch 64部，丟到自己設計的FC (model.py/Net)，經過softmax出來後的label再和Ground Truth label做Cross entropy得到loss。



Discussion:

Training loss的部分算是維持很平穩的下降到後來平穩地在1.5多，而accuracy的部分則波動比較大，但大部分成績也都在0.425以上，取最高0.4603，存為最後的model：best_46.pth

- Report your video recognition performance (valid) using CNN-based video features and make your code reproduce this result. (5%)

```

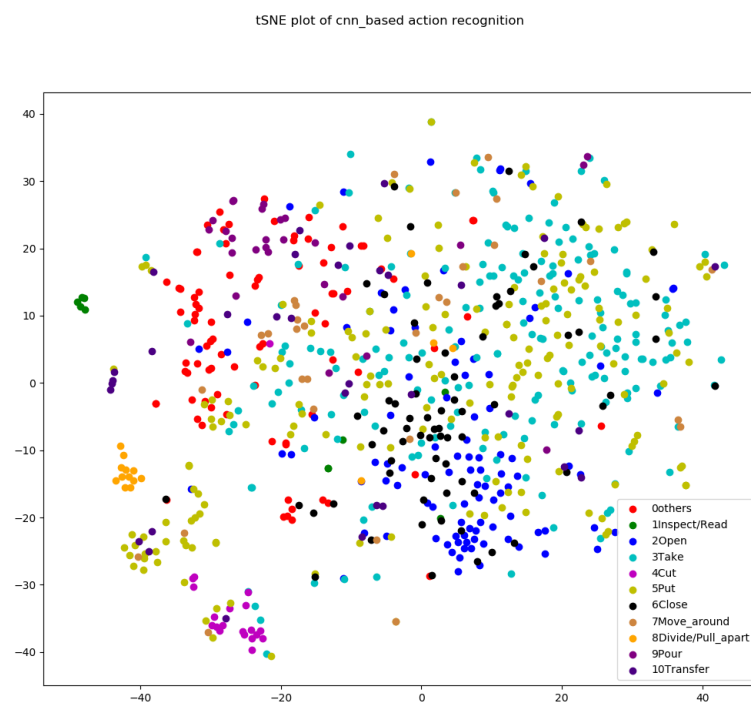
evaluation ans...
predict vals:
[ 2 3 6 5 3 3 5 3 3 3 3 2 3 6 6 5 10 5 5 3 3 3 3 4 4
 4 4 3 3 7 5 6 6 3 2 3 5 10 5 10 5 7 8 8 8 10 8 8 8 8
 8 8 2 8 5 2 2 0 0 5 5 3 2 3 5 5 4 10 7 5 4 4 7 7
 7 7 7 7 4 5 0 4 4 4 5 5 2 2 4 0 3 5 4 4 4 5 6 3
 3 5 5 5 3 3 3 3 2 7 2 5 0 6 7 2 5 2 2 2 3 3 5
 5 5 5 5 3 3 3 3 5 5 7 0 3 5 5 5 7 0 5 5 3 6
 0 0 0 2 3 6 0 2 3 5 5 5 5 5 3 3 6 5 5 5 5 5
 9 9 9 0 0 0 0 0 5 5 7 7 0 0 0 0 5 5 3 3 5 5 5
 5 5 5 2 3 6 3 5 3 3 9 5 3 9 8 9 3 3 5 2 4 5 3 6
 6 3 5 5 5 6 5 2 5 6 5 5 5 0 5 5 5 2 3 6 10 3
 5 5 2 3 3 6 2 3 3 3 3 4 4 4 4 4 4 4 4 4 3 6 3 5
 3 3 4 4 4 4 4 4 4 4 4 5 0 5 9 6 3 3 4 10 3 3 10
 3 10 7 5 5 3 3 7 7 7 7 3 3 4 4 5 7 0 2 6 5 3 6
 5 5 5 7 0 7 5 5 0 6 0 5 9 6 5 5 3 3 5 3 9 7 10 3
 3 9 3 5 5 7 3 10 3 5 3 5 2 3 3 3 3 6 3 5 3 6 5 0
 3 5 3 3 2 6 5 2 2 3 3 3 5 2 3 6 6 3 3 5 3 2 3 2
 3 3 6 3 6 3 6 5 0 6 2 3 5 6 2 2 5 6 5 7 7 7
 7 0 0 0 5 5 7 3 3 5 5 5 2 3 3 5 2 5 2 2 3 2 3
 3 3 5 3 5 3 0 6 3 3 5 1 5 9 0 0 5 0 1 3 2 5 3
 9 2 3 2 3 3 7 9 3 5 5 3 6 2 5 5 5 9 5 5 4 3 3 3
 1 2 2 2 6 3 6 2 2 3 6 2 3 3 5 9 9 9 9 5 0 6 3
 3 3 3 6 6 9 3 3 9 3 3 3 6 0 0 1 3 5 3 5 4 3 3
 3 3 3 9 10 3 3 5 6 5 9 5 6 5 3 2 9 5 5 5 5 9
 0 3 3 6 2 1 6 3 9 5 0 5 3 3 3 3 5 3 3 3 9 6
 3 0 0 5 6 6 2 3 3 2 5 3 2 5 9 5 3 5 5 2 3 5 0 3
 3 5 2 3 2 3 3 5 0 0 4 5 4 2 2 3 3 9 9 5 0 3 2 6
 3 5 5 3 2 5 0 0 3 3 3 2 9 3 3 2 5 6 3 0 0 7 3 6
 3 5 3 0 3 5 2 3 3 2 5 5 2 3 3 5 2 3 3 2 3 0
 3 2 3 3 3 3 3 3 0 3 2 9 5 5 5 5 9 5 2 9 7 2
 5 6 2 9 2 5 5 5 5 5 3 7 7 3 7 3 7 10 10 5 2 5 3
 10 9 6 5 5 3 3 5 5 6 3 6 0 0 8 3 5 3 7 3 2 3
 6 3 3 2 10 10 5 5 5 5 3 3 7 5 5 2 5 3 7 7 3 3 5
5]

label vals:
[ 2 3 5 5 2 3 3 5 2 3 3 2 3 6 6 5 7 5 7 5 2 3 5 4
 4 4 10 5 6 7 5 2 10 10 3 5 10 10 10 5 10 8 8 8 8 8 8
 8 8 8 8 8 2 3 5 5 5 3 10 2 3 5 10 5 5 5 10 5 5 7
 5 5 5 5 10 5 5 5 5 7 5 5 5 5 5 6 3 2 5 5 5 6 3
 5 2 5 5 3 6 6 3 2 3 2 2 2 0 5 0 3 3 6 0 2 5 5
 6 3 5 5 7 3 3 3 3 3 2 2 10 5 3 7 3 2 0 6 5 5
 0 0 0 2 3 3 0 2 3 10 5 2 5 5 3 3 3 6 2 5 3 3
 0 9 3 5 0 10 3 0 10 0 0 0 0 0 0 5 3 5 3 3 5 6
 6 3 6 2 3 3 3 9 5 2 9 6 2 9 6 0 6 5 3 2 5 3 2
 8 5 0 3 9 5 6 6 2 5 6 7 3 5 0 3 3 3 5 2 3 7 3 1
 5 5 2 3 3 6 2 3 2 10 5 3 4 4 4 4 4 4 4 3 5 6 3
 2 10 5 4 10 5 4 4 5 7 4 4 3 0 2 9 6 5 3 3 4 5 6 4
 3 10 5 5 5 3 7 3 7 7 1 1 2 3 5 4 3 4 5 2 2 3 3
 5 2 5 0 0 7 5 3 0 2 0 5 0 5 2 3 2 0 3 0 5 0 3
 3 0 3 10 5 3 3 10 5 3 10 5 2 5 5 5 2 3 3 5 6 5 0
 3 5 6 3 2 5 6 2 2 5 5 3 3 5 5 6 0 3 3 7 3 2 3 2
 2 3 10 0 10 5 3 0 5 5 6 0 2 3 5 3 5 2 3 5 3 0 3
 3 0 0 3 0 0 10 0 0 3 5 3 6 6 5 5 5 6 3 5 2 5
 3 3 3 7 3 10 0 5 7 5 5 1 3 0 9 0 5 5 0 1 3 3 7
 9 6 6 5 5 3 2 0 5 5 5 3 2 3 5 3 9 5 0 3 5 5
 1 1 6 2 6 2 3 6 5 2 10 6 2 5 6 3 0 9 9 5 0 2 3
 10 7 5 6 3 2 0 6 5 0 5 3 2 5 6 0 1 3 2 3 5 3 0 2
 3 2 3 10 0 10 3 3 3 9 5 6 5 3 2 9 6 2 5 0 3 9
 5 3 8 3 5 1 3 2 0 5 3 3 3 3 3 10 6 2 2 2 0 6
 5 0 5 6 0 2 3 1 2 3 5 6 5 9 5 6 6 3 2 5 7 0 5
 3 5 2 3 3 2 6 3 0 0 0 3 6 6 2 5 3 9 9 5 0 2 3 6
 3 9 5 6 2 3 2 9 5 6 3 2 9 5 3 2 5 6 3 7 0 3 9 5
 3 5 7 7 0 5 2 3 9 5 6 3 5 2 3 3 5 3 6 5 5 2 3 0
 0 2 9 6 6 3 5 3 0 0 3 2 9 6 5 0 0 9 5 2 3 6 2
 3 6 2 9 6 2 5 3 5 3 5 7 5 3 5 7 10 0 3 5 2 3
 0 9 6 5 3 2 3 3 5 3 2 5 6 0 0 0 8 0 5 3 7 5 2 3
 6 3 8 3 2 0 5 3 5 7 6 3 7 7 3 2 5 3 7 10 5 0 3 5
7]
accuracy: 0.46033818143042914

```

Validation dataset Acc = 0.4603

- Visualize CNN-based video features to 2D space (with tSNE) in your report. You need to color them with respect to different action labels.(10%)

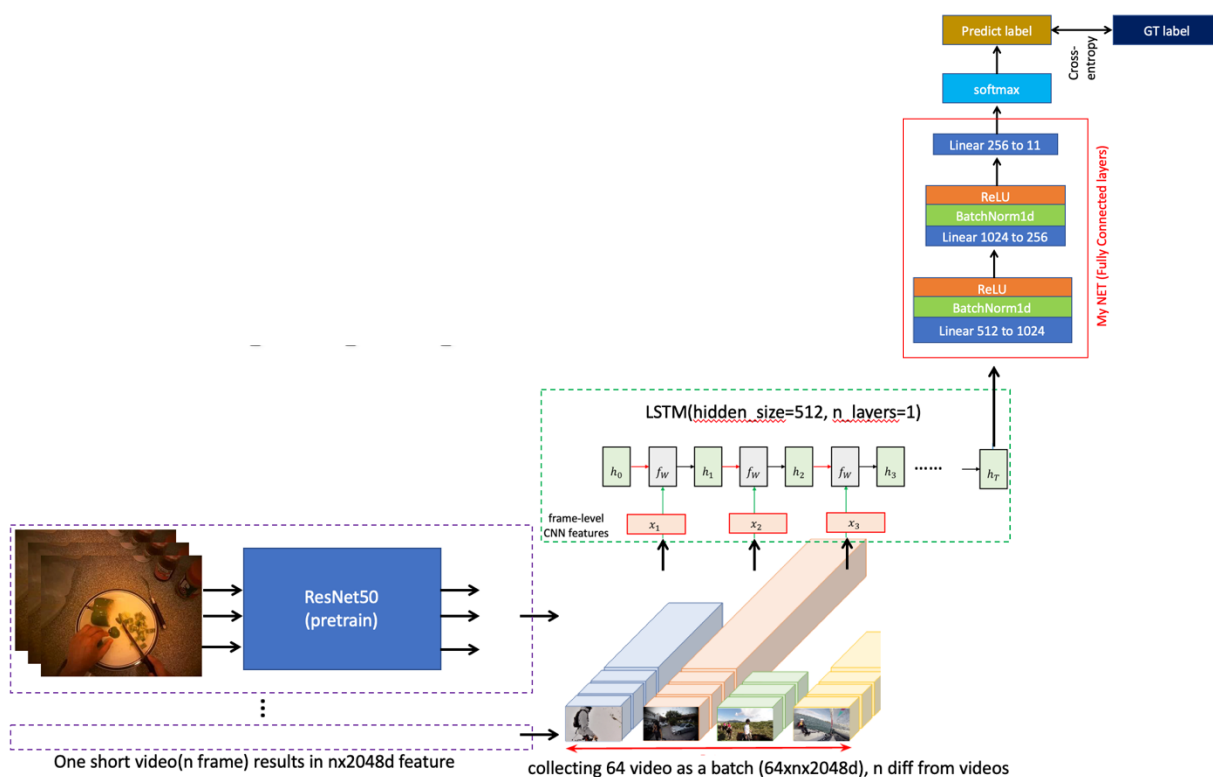


Discussion:

上圖由validation dataset的769部短片，擷取整個架構中最後softmax前的1024d feature，經由tSNE轉換成2d後畫出，以0.46 的accuracy來說，看起來分群的效果算是蠻合理的。

Problem 2 : Trimmed action recognition (40%)

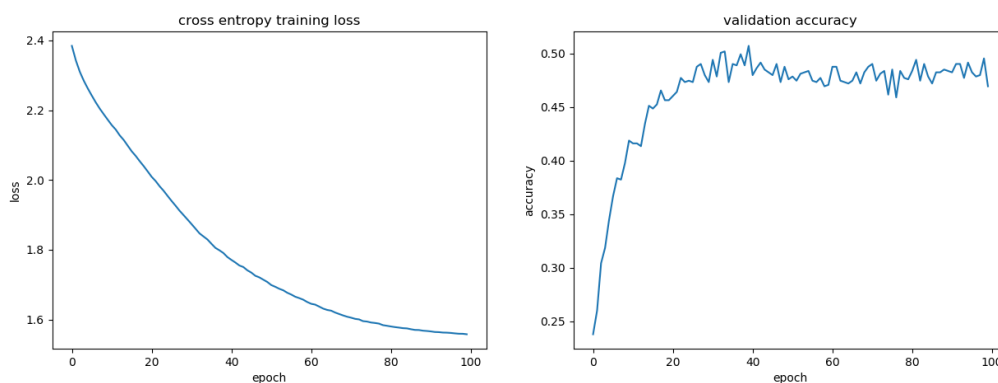
- Describe your RNN models and implementation details for action recognition and plot the learning curve of your model (The loss curve of training set is needed, others are optional). (5%)



Discussion:

這題的training 主要分成兩個步驟，第一個步驟是用pretrain 在ImageNet上的model ResNet50來對p2的所有小短片的每個frame抽出2048d的feature，然後存成一個torch append到list中，所以每支短片得到一個 $n(\text{frame}) \times 2048d$ 的torch，並依序存在list中，最後將此list存成.pt檔。

第二步驟則是再將這個list load近來，並以一個batch 64部做torch.nn.utils.rnn.pad_sequence補齊成大家的n都一樣，變成一包torch，再丟到torch.nn.utils.rnn.pack_padded_sequence，變成一包PackedSequence object再進LSTM，取出最後的512d hidden state再丟，自己設計的FC (model.py/Net)，經過softmax出來後的label再和Ground Truth label做Cross entropy得到loss。



Training的結果，loss正常，accuracy=50.7則比p1純cnn的方法高，曲線也較穩。

- Your model should pass the baseline (valid: 0.45 / test: 0.43) validation set (10%) / test set (15%, only TAs have the test set).

```

evaluation ans...
predict vals:
[ 2 3 5 5 3 3 5 2 3 3 2 3 6 6 5 3 3 3 3 3 3 3 10
 4 4 7 7 5 5 3 3 2 5 2 5 10 3 10 3 3 3 4 3 3 3
 3 4 6 5 5 2 5 0 2 5 3 3 3 3 5 2 5 3 3 4 5 5 2
 5 5 5 5 5 5 5 5 5 5 5 5 5 5 2 5 4 2 3 5 2 3
 3 5 5 5 3 3 2 2 2 2 5 0 5 0 4 5 0 2 3 5 2 5
 6 5 5 5 5 3 3 3 3 5 5 3 10 5 5 5 3 7 0 5 5 1 2
 5 0 0 2 5 6 6 2 2 5 5 3 5 5 3 3 6 3 5 3 3 3
 0 0 0 0 0 0 5 0 5 3 0 0 0 0 10 5 5 5 5 3 3 3
 5 5 5 2 3 6 5 3 2 0 5 5 5 5 0 3 5 5 2 3 6 3 3
 2 3 0 3 10 5 6 0 2 5 6 5 5 5 0 5 5 5 2 3 0 3 3
 2 5 2 5 3 6 2 5 2 3 5 5 4 4 3 4 4 4 4 5 2 3 5
 2 5 5 4 4 3 4 4 5 4 4 5 0 5 10 5 5 3 4 4 3 5 4
 3 10 3 5 5 3 4 3 7 5 6 0 2 5 4 4 5 2 3 3 5 6
 5 5 3 7 0 7 5 5 0 3 3 5 5 5 2 3 3 0 3 10 10 3
 3 4 3 7 5 5 3 10 10 5 10 0 2 5 5 5 5 0 3 5 3 6 5 0
 5 5 5 3 2 6 6 2 2 3 3 5 5 2 5 6 7 3 3 5 3 3 5 2
 3 3 0 2 0 3 3 0 2 6 6 6 2 3 5 5 2 2 5 3 2 2 3 3
 3 0 0 5 5 2 2 2 0 5 3 3 2 3 5 5 2 5 3 2 2 2 3
 3 3 5 5 4 0 3 3 4 1 3 0 0 0 5 5 0 1 3 3 3 3
 10 6 6 2 5 3 6 0 6 5 5 3 5 5 5 7 5 0 5 3 3 3
 1 2 6 2 2 3 6 5 2 5 6 5 5 3 5 0 10 0 0 5 3 3
 10 3 3 2 3 2 10 3 3 10 3 3 6 6 6 6 1 5 3 5 3 10 5
 3 3 3 10 10 0 4 10 3 3 3 0 3 6 5 3 2 5 7 5 0
 0 3 2 2 2 1 3 2 0 5 5 5 3 5 3 5 3 6 3 3 2 5
 3 0 0 5 6 0 2 3 6 2 3 5 2 5 10 5 5 5 2 3 3 0 3
 3 5 2 3 3 3 3 5 5 0 5 0 3 2 3 3 0 0 5 0 3 3 1
 3 2 0 2 5 6 0 5 3 2 0 3 2 5 3 5 0 3 10 5
 3 5 3 5 5 2 3 5 2 6 0 5 2 3 3 5 5 6 3 3 3 0
 0 2 5 3 3 3 3 0 3 2 5 0 3 3 0 3 5 3 3 3 2
 3 6 3 3 2 5 5 5 3 7 3 3 3 3 3 3 3 5 2 3 3
 10 5 3 3 3 4 5 0 5 2 5 3 0 0 3 5 3 7 3 2 3
 6 3 3 0 0 5 3 3 5 5 3 7 5 3 2 5 3 7 10 7 3 3 5
 5]

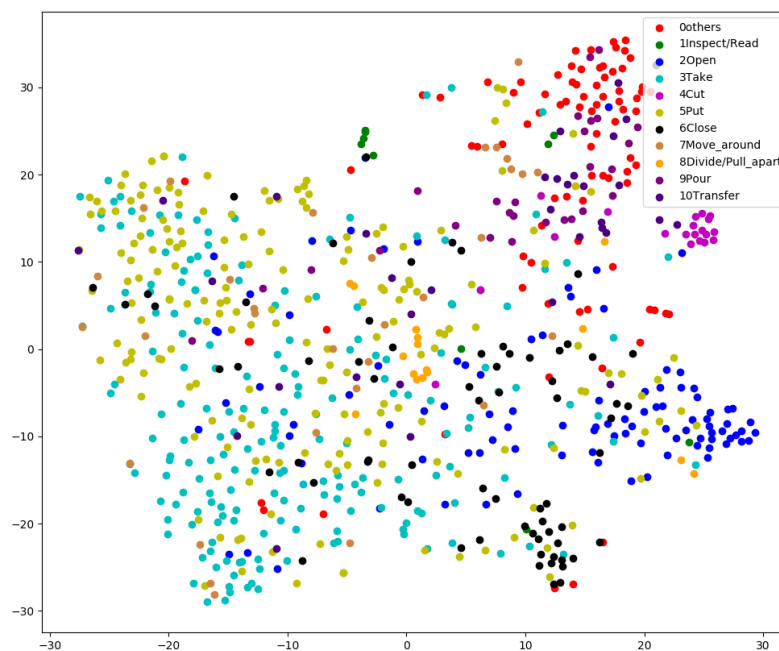
Label_vals:
[ 2 3 5 5 2 3 3 5 2 3 3 2 3 6 6 5 7 5 7 5 2 3 5 4
 4 4 10 5 6 7 5 2 10 10 3 5 10 10 5 10 8 8 8 8 8 8
 8 8 8 8 8 2 3 5 5 5 3 10 2 3 5 10 5 5 10 5 5 7
 5 5 5 10 5 5 5 5 7 5 5 5 5 6 3 2 5 5 5 6 3
 5 2 5 3 6 3 2 3 2 2 0 5 0 3 6 0 2 5 5 5
 6 3 5 5 7 3 3 3 3 2 2 3 10 5 3 7 3 2 0 6 5 2 5
 0 0 0 2 3 3 0 2 3 10 5 2 5 5 3 3 3 6 2 5 3 5 3
 0 9 3 5 0 10 3 0 10 0 0 0 0 0 0 5 3 5 5 3 5 6
 6 3 0 2 3 3 3 9 5 2 9 6 2 9 6 0 6 5 3 2 5 3 2
 8 5 0 3 9 5 6 6 2 5 6 7 3 5 0 3 3 5 2 3 7 3 1
 5 5 2 3 3 6 2 3 2 10 5 3 4 4 4 4 4 4 3 5 6 3
 2 10 5 4 10 5 4 4 5 7 4 4 3 0 2 9 6 5 3 3 4 5 6 4
 3 10 5 5 5 3 7 3 7 7 1 1 2 3 5 4 3 4 5 2 2 3 3 3
 5 2 5 0 0 7 5 3 0 2 0 5 0 5 2 3 5 2 3 0 5 0 3
 3 0 3 10 5 3 3 10 5 3 10 5 2 5 5 5 2 3 3 5 6 5 0
 3 5 6 3 2 5 6 2 2 5 5 3 3 5 6 0 3 3 7 3 2 3 2
 2 3 10 0 10 5 3 0 5 5 6 0 2 3 5 3 2 3 5 3 0 3
 3 0 0 3 0 0 10 0 0 3 5 3 6 6 5 5 5 6 2 3 2 5
 3 3 3 7 3 10 0 5 7 5 5 1 3 0 9 0 5 5 0 1 3 3 3 7
 9 6 6 5 5 3 2 0 5 5 5 3 2 3 5 5 3 9 5 0 3 5 5 5
 1 1 6 2 6 2 3 6 5 2 10 6 2 5 6 3 0 9 0 9 5 0 2 3
 10 7 5 6 3 2 0 6 5 0 5 3 2 5 6 0 1 3 2 3 5 3 0 2
 3 2 3 10 0 10 3 0 3 3 9 5 6 5 3 2 9 6 2 5 0 3 9
 5 3 8 3 5 1 3 2 0 5 3 5 3 3 3 3 10 6 2 2 2 0 6
 5 0 0 5 6 0 2 3 1 2 3 5 6 5 9 5 6 3 2 5 7 0 5
 3 5 2 3 3 2 0 3 0 0 0 3 6 6 2 5 3 9 9 5 0 2 3 6
 3 9 5 6 3 2 0 5 6 3 2 9 5 3 2 5 6 3 7 0 3 9 5
 3 5 7 7 0 5 2 3 9 5 6 3 5 2 3 5 6 5 2 3 0
 0 2 9 6 6 3 3 5 3 0 0 3 2 9 6 5 0 9 5 2 3 6 2
 3 6 2 9 6 2 5 3 5 3 5 7 5 3 7 3 10 0 3 5 2 3 3
 0 9 6 5 3 2 3 3 5 3 2 5 6 0 0 0 8 0 5 3 7 5 2 3
 6 3 8 3 2 0 5 3 5 7 6 3 7 7 3 2 5 3 7 10 5 0 3 5
 7]
accuracy: 0.5071521456436932

```

Validation dataset Acc = 0.5072

- Visualize RNN-based video features to 2D space (with tSNE) in your report. You need to color them with respect to different action labels. Do you see any improvement for action recognition compared to CNN-based video features ? Why? Please explain your observation (10%).

tSNE plot of cnn_based action recognition

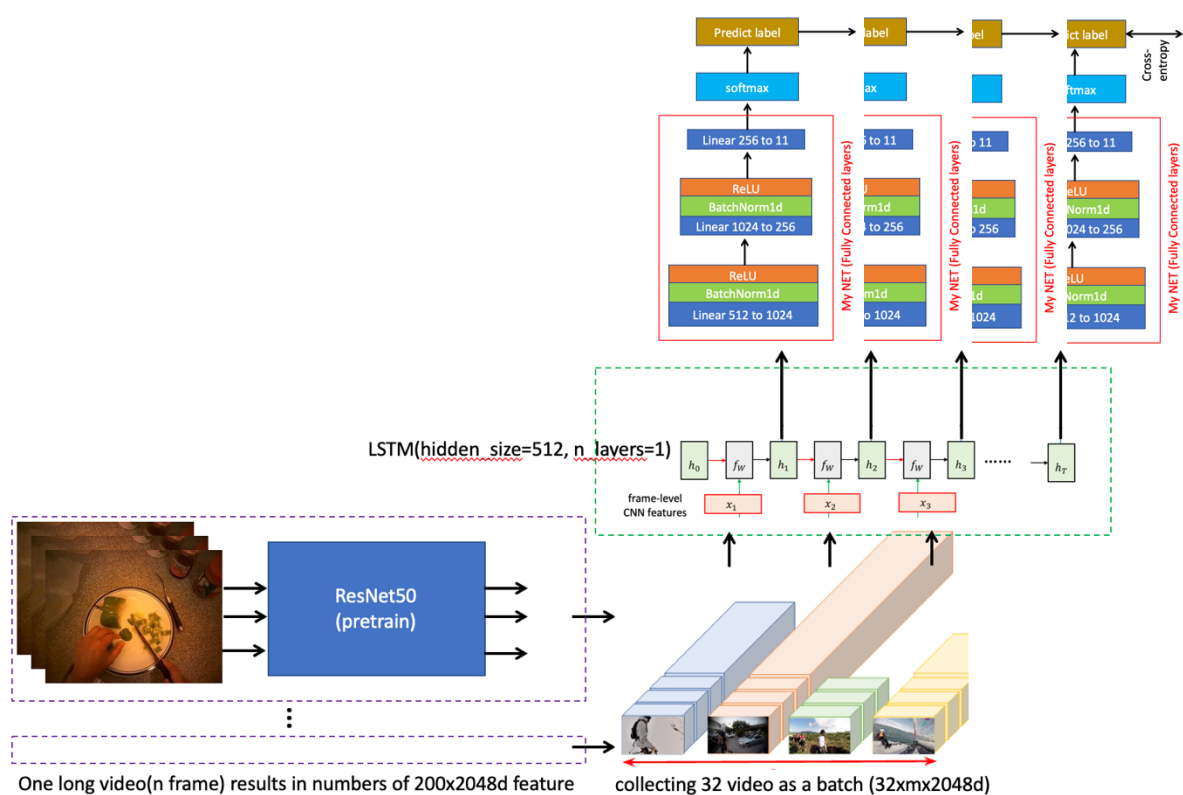


Discussion:

上圖由validation dataset的769部短片，擷取整個架構中最後softmax前的256dfeature，經由tSNE轉換成2d後畫出，相較前一個純cnn的方法，accuracy高了快5%左右，分群的效果看起來算是合理，但和前者p1的差別不至於太大(畢竟只高了5%)但以training的想法來說，p2的train法使用lstm具有frame和frame之間記憶傳遞的效果，對於我們的目的是action recognition要看的也是個連續的過程來說，比起p1的直接平均，較合理一些，所以accuracy也較高。

Problem 3 : Seq-to-Seq prediction in full-length videos (40%)

- Describe any extension of your RNN models, training tricks, and post-processing techniques you used for temporal action segmentation. (5%)



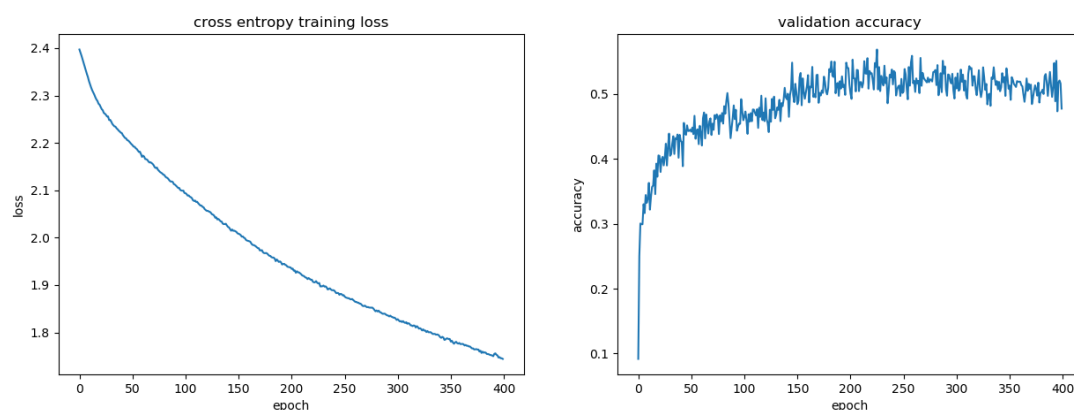
Discussion:

這題的training 主要分成兩個步驟，第一個步驟是用pretrain 在ImageNet上的model ResNet50來對p3的所有常片的每個frame抽出2048d的feature，然後200個frame成一單位，存成一個小torch append到list中，所以每支長片得到 $\text{ceil}(n \text{ frames}/200)$ 部短片，該短片存成 200(但最後一部未滿200) x 2048d 的torch，並依序存在list中，最後將此list存成.pt檔。

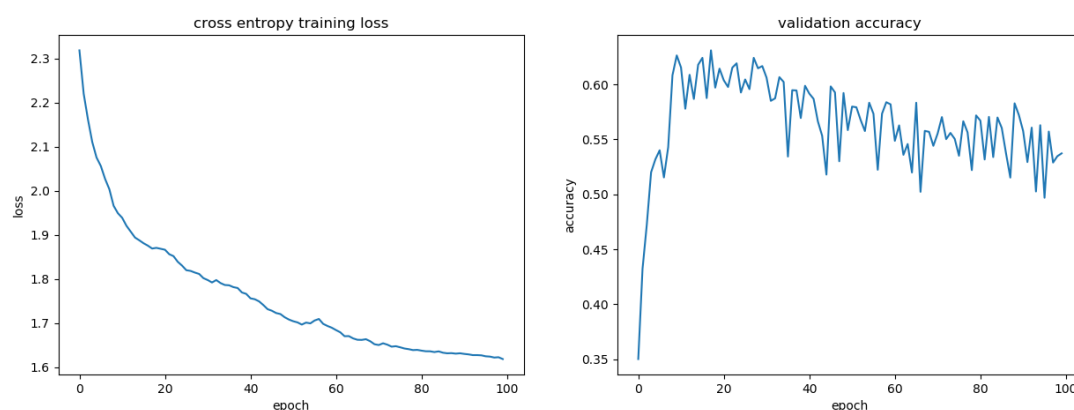
第二步驟則是再將這個list load近來，並以一個batch 32部做 `torch.nn.utils.rnn.pad_sequence` 補齊成大家的n都一樣，變成一包torch，再丟到 `torch.nn.utils.rnn.pack_padded_sequence`，變成一包PackedSequence object再進LSTM，取出所有512d hidden state再丟，自己設計的FC (model.py/Net)，經過softmax出來後的label再和Ground Truth label做Cross entropy得到loss。其中要特別注意的是，因為長片都被以200 frame一刀切成短片，所以再算loss的時候要對回原本的label要特別注意順序有沒有跑掉，只要順序對了，acc就能勸超過50%。

而下圖為這題的training loss 和 accuracy，loss大致上沒問題，acc的部分，一開始嘗試了不使用p2 train好的weight來initialize，直接開train，得到了0.528的accuracy並且波型還算穩定。後來使用p2 train好的weight來initialize，得到了0.587的accuracy但波型相較前者沒這麼穩定了，後續會再討論者兩者的預測品質的問題。

Seq2seq w/o p2 pretrain weight



Seq2seq w/o p2 pretrain weight



- Report validation accuracy in your report and make your code reproduce this result.

(20%)

(accuracy show in below table)

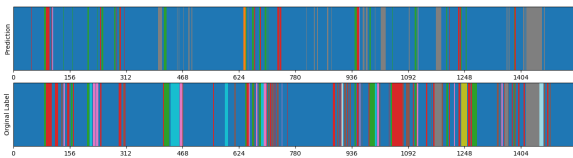
- Choose one video from the 7 validation videos to visualize the best prediction result in comparison with the ground-truth scores in your report. Please make your figure clear and explain your visualization results (You need to plot at least 500 continuous frames). (15%)

Using `matplotlib.colorbar.ColorbarBase` to plot the visualizer, and choosing color by color map in `matplotlib.pyplot`. (“tab20”)

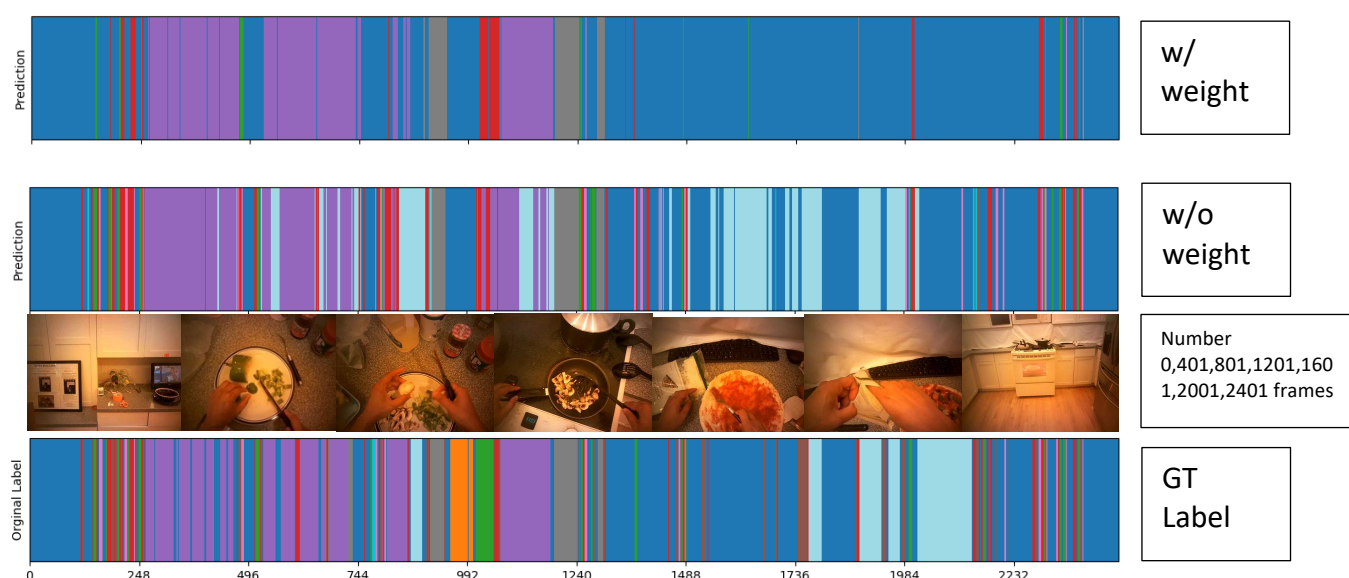


others	Inspect/ Read	open	take	cut	put	close	Move around	Divide/ pull apart	pour	transfer
--------	------------------	------	------	-----	-----	-------	----------------	--------------------------	------	----------

(下表中上側のcolorbar為predict, 下側の為GT label)

	seq2seq w/ p2 pretrain weight	seq2seq w/o p2 pretrain weight
Individual accuracy and mean accuracy	<p>OP01-R02-TurkeySandwich.txt evaluation ans... accuracy: 0.4733201581027668</p> <p>OP01-R04-ContinentalBreakfast.txt evaluation ans... accuracy: 0.6038696537678208</p> <p>OP01-R07-Pizza.txt evaluation ans... accuracy: 0.6402266288951841</p> <p>OP03-R04-ContinentalBreakfast.txt evaluation ans... accuracy: 0.5410573678290214</p> <p>OP04-R04-ContinentalBreakfast.txt evaluation ans... accuracy: 0.6571428571428571</p> <p>OP05-R04-ContinentalBreakfast.txt evaluation ans... accuracy: 0.5305907172995781</p> <p>OP06-R03-BaconAndEggs.txt evaluation ans... accuracy: 0.6634429400386848</p> <p>mean_accuracy: 0.5870929032965589</p>	<p>OP01-R02-TurkeySandwich.txt evaluation ans... accuracy: 0.45948616600790515</p> <p>OP01-R04-ContinentalBreakfast.txt evaluation ans... accuracy: 0.5468431771894093</p> <p>OP01-R07-Pizza.txt evaluation ans... accuracy: 0.5770942938081748</p> <p>OP03-R04-ContinentalBreakfast.txt evaluation ans... accuracy: 0.4881889763779528</p> <p>OP04-R04-ContinentalBreakfast.txt evaluation ans... accuracy: 0.5557603686635945</p> <p>OP05-R04-ContinentalBreakfast.txt evaluation ans... accuracy: 0.520042194092827</p> <p>OP06-R03-BaconAndEggs.txt evaluation ans... accuracy: 0.5499677627337202</p> <p>mean_accuracy: 0.5281975626962263</p>
OP01-R02-Turkey Sandwich.txt		
OP01-R04-Continental Breakfast.txt		
OP01-R07-Pizza.txt		
OP03-R04-Continental Breakfast.txt		
OP04-R04-Continental Breakfast.txt		
OP05-R04-Continental Breakfast.txt		
OP06-R03-Bacon AndEggs.txt		

Best result: OP01-R07-Pizza.txt (acc= 0.640 w/ weight, 0.577 w/o weight)



Discussion:

由上方結果可知，因為 GT label 的資料中，本來就很多是 others，甚至如果整條都 predict 成 others(整條都藍色的)，accuracy 也會有 40 幾 (聽說的)，因此本題中的 visualizer 就可以拿來仔細檢閱 predict 出來的品質。

可以發現 Seq2seq w/o p2 pretrain weigh 預測出的 label 多樣性明顯比 Seq2seq w/ p2 pretrain weight 的好，雖然錯的比較多(acc 較低)，但不會都是藍色(others)，或許是因為 load 進來的 weight 因為之前已經 train 過知道結果會偏向 others，因此以這個 weight 起手出來的結果也會偏預測 others 居多。

另外就是不同部的長片中屬第一部的accuracy是七部中的最低，是因為本人在本次作業中並沒有將loss寫得很完美，因為丟進去train的長片已經被切散成最大200 frame的長度，並將不足的做0 padding補齊，預測完吐出來之後也是有padding的結果，在沒有查到很好解決de-padding的方法之下，我將GT label也照同樣的切法並且padding 0 補齊，再將兩者做cross-entropy。因此觀察各個長片的frame數 [1012 982 2471 889 1085 948 1551]，就屬第一部影片用200 frame去切會切出只有12 frame其他188 frame都是靠padding來的結果，導致不管那12個frame預測的多爛，loss都會偏低，因為其他靠padding來的frame預測一定是準的。

Bonus: Attention mechanisms (up to 20%)