Name: 湯忠憲　Dep.:資料科學碩一　Student ID:R06946003

[Problem1]

1. **Describe your strategies of extracting CNN-based video features, training the model and other implementation details.**

   DenseNet 121 pre-trained on ImageNet is adapted to extract the CNN-based video features. To meet the requirement of pre-trained model, a single rectangle video frame would be padded to a square, normalized and resized to shape (224,224). After the preprocessing, the image would go through 4 DenseBlock then a feature map with shape (1024, 7, 7) is generated. For feature extraction, the feature map is flattened to shape (50176,). To represent a video feature using a vector, all flattened-frame level features basically follow an average-pooling.

   A video recognition network, as shown in Fig. 1, consists of three fully connected layers with Relu activation function. The loss function is cross-entropy and the optimizer is Adam with 0.0001 learning rate.

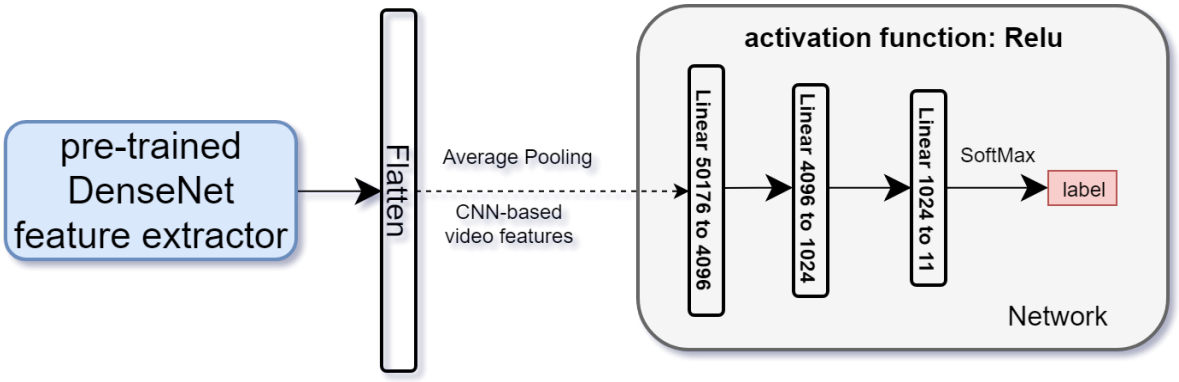


Fig. 1. Video action recognition pipeline

2. **Report your video recognition performance using CNN-based video features and plot the learning curve of your model.**

   The video recognition performance could reach 0.49 accuracy on validation set using 4 fps. Fig. 2 shows the training loss and validation accuracy for each epoch.

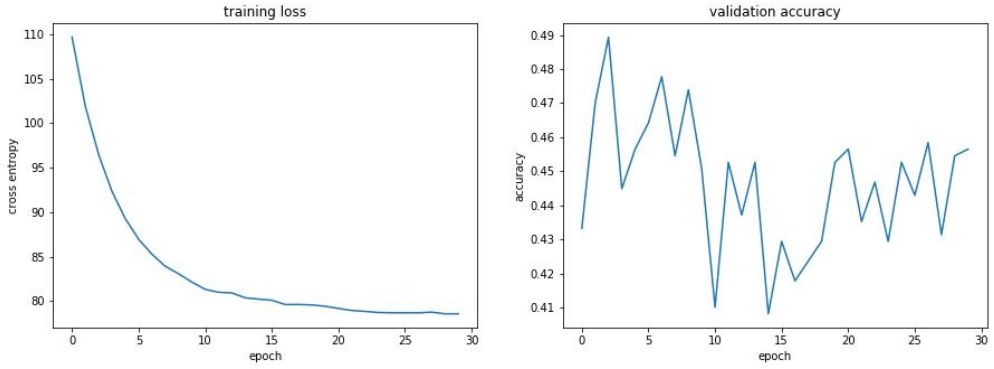


Fig. 2. The learning curve

[Problem2]

1. **Describe your RNN models and implementation details for action recognition.**

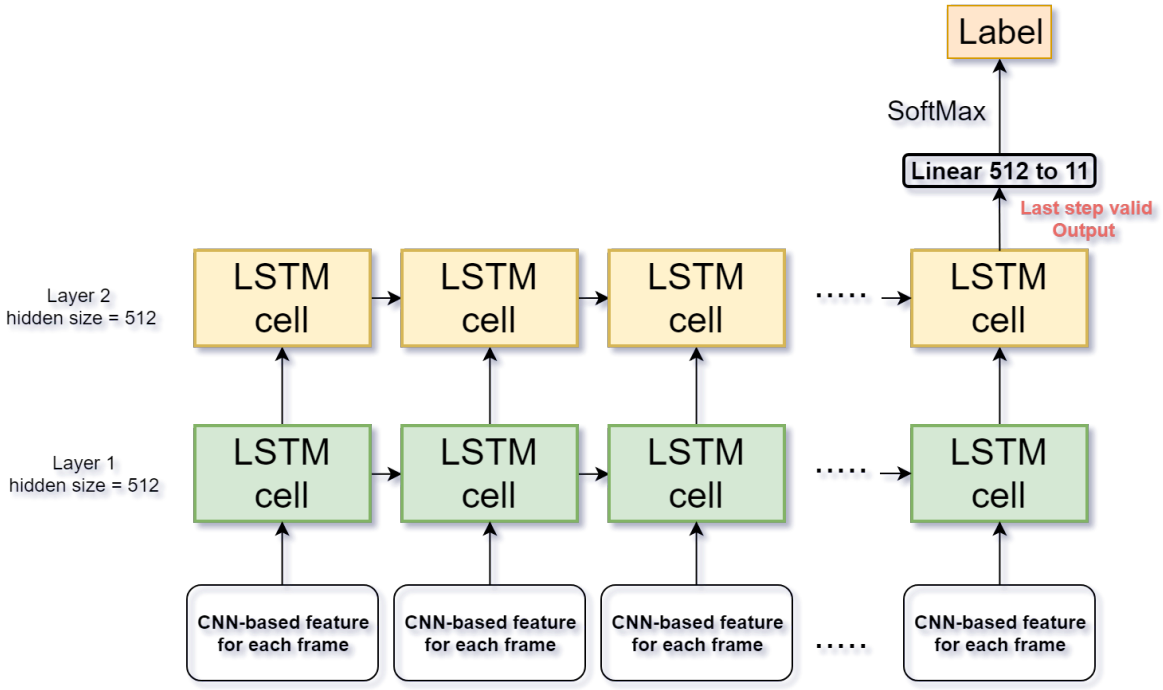


Fig. 3. RNN models for action recognition

The RNN model consists of two LSTM layer with hidden size 512. A frame-level feature is fed into the LSTM cell each timestep. For a fix training length, each batch of video features are zero padded[1] and take valid output only. The last valid output flow through a fully connected layer with Softmax function to generate prediction label. The training curve and validation accuracy are shown below. The model could reach about 0.5 accuracy on validation set within 5 epochs.

The loss function is cross-entropy, optimizer is Adam with 0.0001 learning rate and batch size is 64.

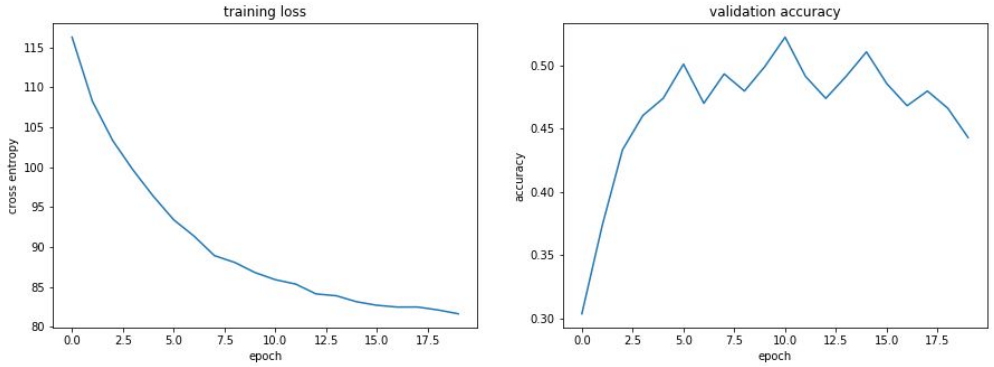
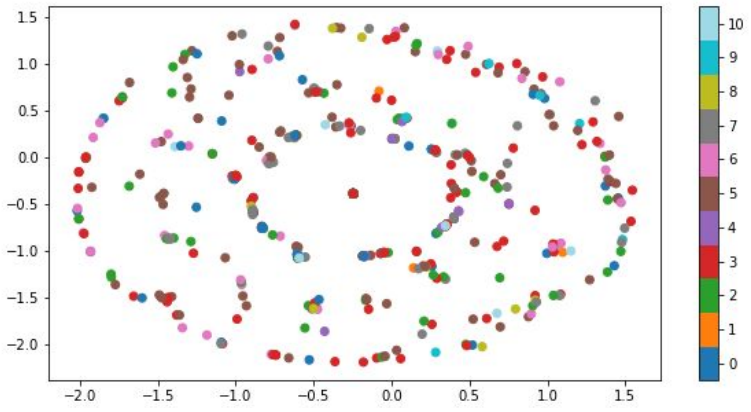


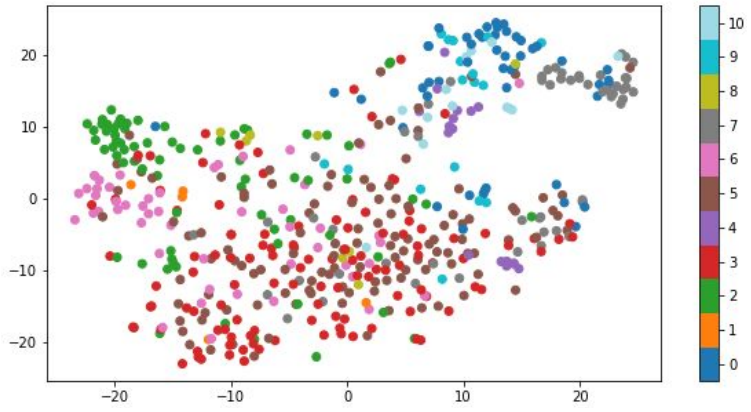
Fig. 4. The learning curve

[1] https://zhuanlan.zhihu.com/p/34418001

2. **Visualize CNN-based video features and RNN-based video features to 2D space (with tSNE). You need to generate two separate graphs and color them with respect to different action labels. Do you see any improvement for action recognition? Please explain your observation.**



(a) CNN-based features



(b) RNN-based features

Fig. 5. Visualization of (a) CNN feature and (b) RNN feature

Table 1. Action labels

0	1	2	3	4	5	6	7	8	9	10
Other	Inspect /Read	Open	Take	Cut	Put	Close	Move Around	Divide/ Pull Apart	Pour	Transfer

Two separate graphs for CNN and RNN feature visualization are shown in Fig. 5 and the corresponding actions are illustrated in Table 1. According to the observation, the CNN features, represented by different color dot, seem to be mixd up in the 2-dimensional plot and hard to discover patterns. One reason for that is the dimension of CNN-based feature is very high here. On the other hand, some of the action labels group together in the RNN feature visualization, e.g., *Open*, *Other* and *Take*. The model does learn the representation of input frames. Therefore, RNN model could generate better results.

[Problem3]

1. **Describe any extension of your RNN models, training tricks, and post-processing techniques you used for temporal action segmentation.**

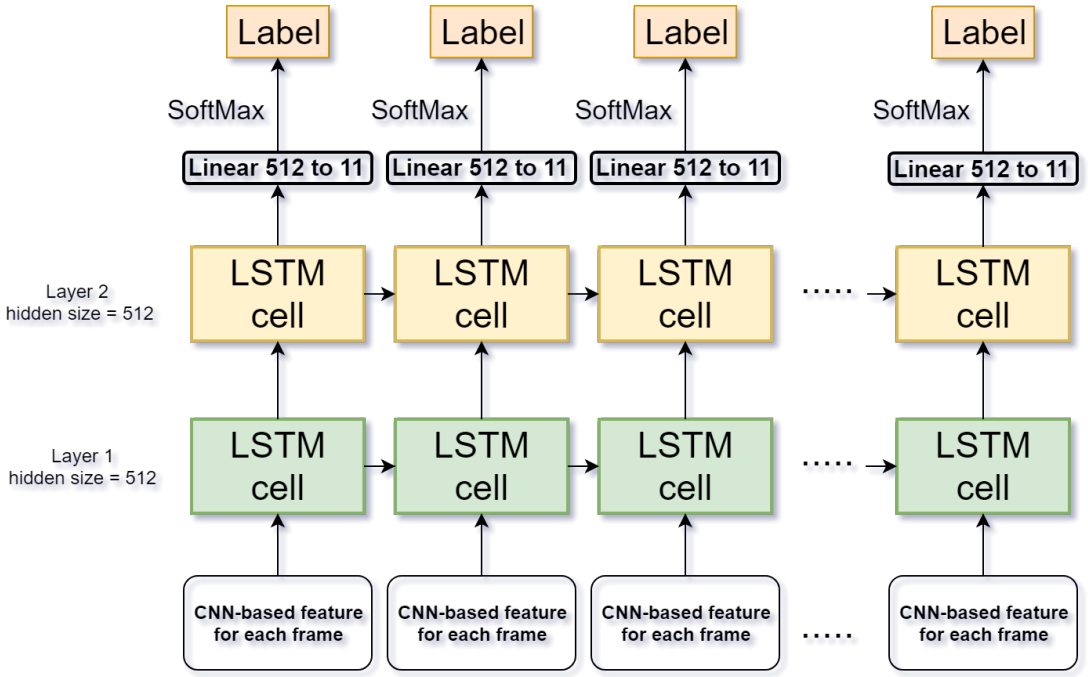


Fig. 6. RNN models for temporal action recognition

The model for temporal action recognition is an extension from previous RNN model, which simply outputs a prediction every time step, as shown in Fig. 6. For training instances, all frames from a single video are cut to suitable training size with frames overlap to the previous frame segment. For example, we can assign 300 frames for a training instance, and the first 30 frames are same as the last 30 frames of the previous frame segment (in the same video). The process is illustrated in Fig. 7.

In terms of the hyper-parameter, the hidden size is 512 for both LSTM layers with 0.5 dropout rate, the optimizer is Adam with 0.0001 learning rate, and the batch size is 32.

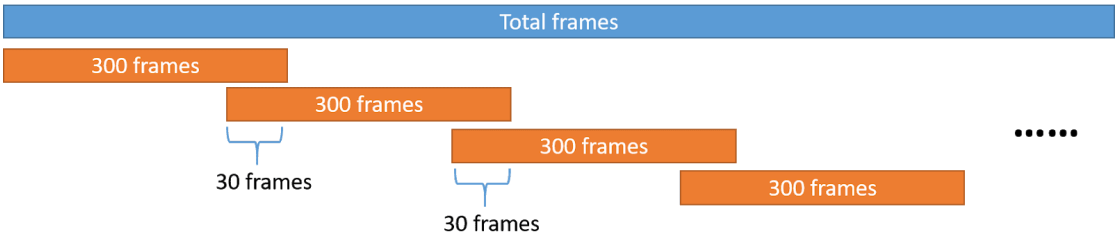


Fig. 7. Frame segmentation

2. **Report validation accuracy and plot the learning curve.**

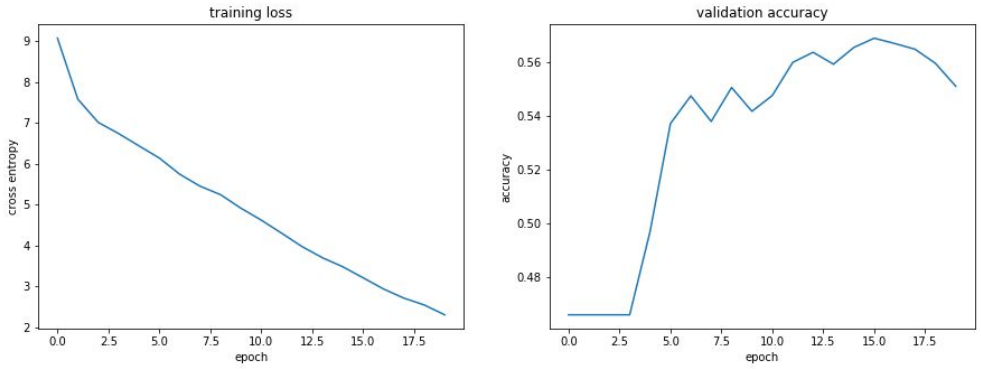


Fig. 8. The learning curve

The model generates predictions with all zeros (*Other*) for the first 3 epochs on validation set. Since there are lots of *Other* in training and validation ground-truth, the validation accuracy for first 3 epochs could reach 46% with those zero-predictions, as shown in Fig.8. Fortunately, after several iterations, the validation accuracy for all frames turns into 0.57.

3. **Choose one video from the 5 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 300 continuous frames (2.5 mins).**

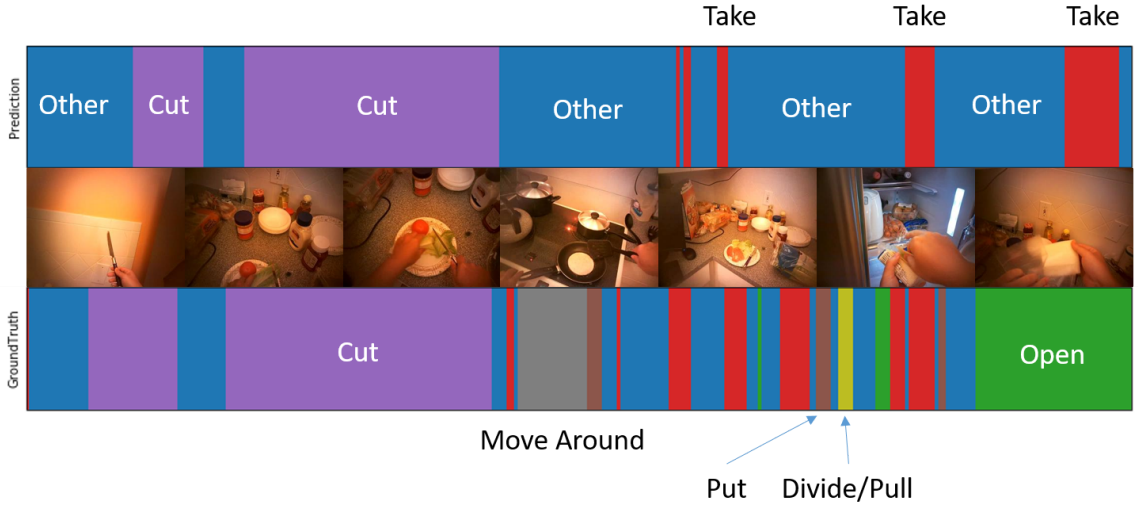
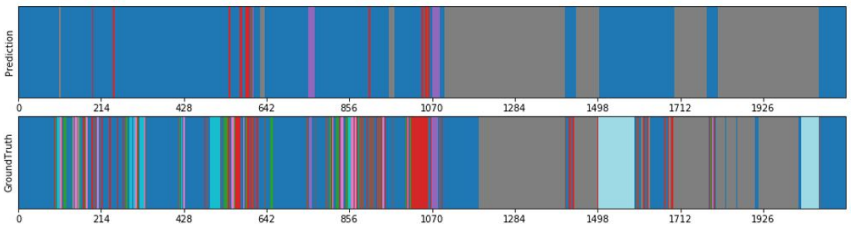
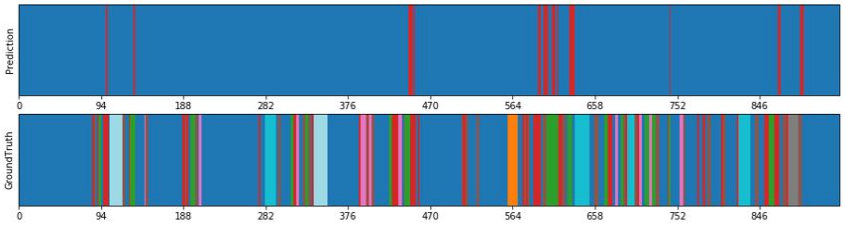
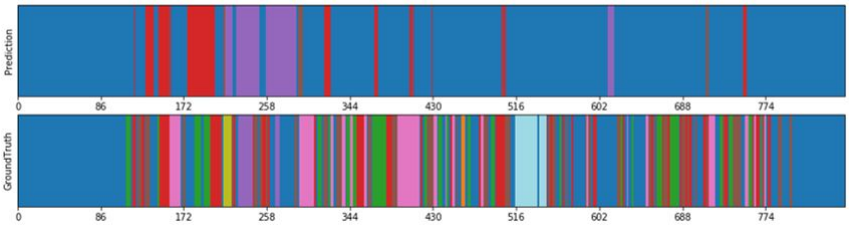
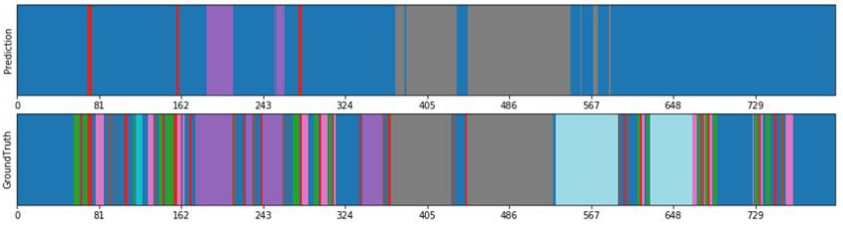


Fig. 9. Visualization of temporal action segmentation

Visualization of temporal action segmentation with 300 frames from video *OP06-R05-Cheeseburger* is depicted in above figure. *Other* and *Cut* are more accurate comparing to other actions since they have lots of training instances. The following table displays predictions for 5 validation videos. According to the results, actions with more training samples did receive greater performance.

Video name	Visualization
Bacon And Eggs	
Continental Breakfast	
Turkey Sandwich	
Pizza	
Cheeseburger