Problem 2 task a

Thought: I have to use unsupervised learning approach to solve this problem. Because I have to train video which are not anomaly. For this purpose, I want to use autoencoder algorithm. As autoencoder can be used for reconstruction of input data as close to original input. By using this concept, I can easily solve this problem.

Steps of approach:

1. Find out the dataset which contains normal clip and anomaly clip of specific location.
2. Resize each frame into specific size
3. Perform data augmentation if needed.
4. Perform scaling by dividing 255 of each pixel of each frame
5. Design autoencoder model
6. Train the model and tune the hyper parameters.
7. Evaluate the model by measuring reconstruction score.

The Steps I followed to solve this model:

1. I used UCSD dataset. This dataset contains two location footage, Peds1 and Peds2.
2. I used Peds1 clips in which a groups of people walking in a way in font of camera. This contains 34 training video and 36 testing (anomaly and normal) video. Each video contains 200 frames.
3. I resized each frame to 256 * 256 size.
4. As this is not enough for deep learning model I used data augmentation to increase the data.
5. For data augmentation I used sliding window technique.
6. I scaled each frame pixel between 0 to 1 by diving 255.
7. For model creation I used convolutional lstm auto encoder algorithm.
8. The model summary is given bellow:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
time_distributed (TimeDistri (None, 5, 64, 64, 64)     3200
_________________________________________________________________
layer_normalization (LayerNo (None, 5, 64, 64, 64)     128
_________________________________________________________________
time_distributed_1 (TimeDist (None, 5, 32, 32, 32)     51232
_________________________________________________________________
layer_normalization_1 (Layer (None, 5, 32, 32, 32)     64
_________________________________________________________________
conv_lst_m2d (ConvLSTM2D)    (None, 5, 32, 32, 32)     73856
_________________________________________________________________
layer_normalization_2 (Layer (None, 5, 32, 32, 32)     64
_________________________________________________________________
conv_lst_m2d_1 (ConvLSTM2D)  (None, 5, 32, 32, 16)     27712
_________________________________________________________________
layer_normalization_3 (Layer (None, 5, 32, 32, 16)     32
_________________________________________________________________
conv_lst_m2d_2 (ConvLSTM2D)  (None, 5, 32, 32, 32)     55424
_________________________________________________________________
layer_normalization_4 (Layer (None, 5, 32, 32, 32)     64
_________________________________________________________________
time_distributed_2 (TimeDist (None, 5, 64, 64, 32)     25632
_________________________________________________________________
layer_normalization_5 (Layer (None, 5, 64, 64, 32)     64
_________________________________________________________________
time_distributed_3 (TimeDist (None, 5, 256, 256, 64)   100416
_________________________________________________________________
layer_normalization_6 (Layer (None, 5, 256, 256, 64)   128
_________________________________________________________________
time_distributed_4 (TimeDist (None, 5, 256, 256, 1)    3137
=================================================================
Total params: 341,153
Trainable params: 341,153
Non-trainable params: 0
```

9. I ran this model over 20 epochs. And measure mse score of each epoch.
10. As model contains huge parameters I used nvidia 1650 GPU.
11. The tarin results is given bellow:

```
Train on 1360 samples
Epoch 1/20
1360/1360 [==============================] - 241s 177ms/sample - loss: 0.0082 - mse: 0.0082
Epoch 2/20
1360/1360 [==============================] - 233s 171ms/sample - loss: 0.0032 - mse: 0.0032
Epoch 3/20
1360/1360 [==============================] - 233s 172ms/sample - loss: 0.0025 - mse: 0.0025
Epoch 4/20
1360/1360 [==============================] - 235s 173ms/sample - loss: 0.0021 - mse: 0.0021
Epoch 5/20
1360/1360 [==============================] - 233s 172ms/sample - loss: 0.0018 - mse: 0.0018
Epoch 6/20
1360/1360 [==============================] - 233s 172ms/sample - loss: 0.0016 - mse: 0.0016
Epoch 7/20
1360/1360 [==============================] - 233s 172ms/sample - loss: 0.0015 - mse: 0.0015
Epoch 8/20
1360/1360 [==============================] - 234s 172ms/sample - loss: 0.0014 - mse: 0.0014
Epoch 9/20
1360/1360 [==============================] - 233s 172ms/sample - loss: 0.0012 - mse: 0.0012
Epoch 10/20
1360/1360 [==============================] - 234s 172ms/sample - loss: 0.0012 - mse: 0.0012
Epoch 11/20
1360/1360 [==============================] - 233s 172ms/sample - loss: 0.0011 - mse: 0.0011
Epoch 12/20
1360/1360 [==============================] - 233s 172ms/sample - loss: 0.0010 - mse: 0.0010
Epoch 13/20
1360/1360 [==============================] - 234s 172ms/sample - loss: 9.8769e-04 - mse: 9.8769e-04
Epoch 14/20
1360/1360 [==============================] - 234s 172ms/sample - loss: 9.3723e-04 - mse: 9.3723e-04
Epoch 15/20
1360/1360 [==============================] - 234s 172ms/sample - loss: 9.0167e-04 - mse: 9.0167e-04
Epoch 16/20
1360/1360 [==============================] - 234s 172ms/sample - loss: 8.6828e-04 - mse: 8.6828e-04
Epoch 17/20
1360/1360 [==============================] - 233s 172ms/sample - loss: 8.3453e-04 - mse: 8.3453e-04
Epoch 18/20
1360/1360 [==============================] - 233s 172ms/sample - loss: 8.0547e-04 - mse: 8.0547e-04
Epoch 19/20
1360/1360 [==============================] - 233s 172ms/sample - loss: 7.7829e-04 - mse: 7.7829e-04
Epoch 20/20
1360/1360 [==============================] - 234s 172ms/sample - loss: 7.5343e-04 - mse: 7.5343e-04
```

12. Above figure represents my model performed very good. As mse decreasing gradually.
13. After train the model I save it as h5 format.

Model Evaluation:

1. For evaluation I used anomaly data.
2. I preprocessed each frame as the same way I preprocessed training data.
3. Then I passed it to model for making prediction.
4. Model return me reconstructed sequence
5. From reconstructed sequence and calculated reconstruction cost of each frame.
6. Then by subtracting from 1 I founded reconstruction score.
7. I calculated the mean of reconstruction score. If the reconstruction score is below the average, I marked this frame as anomaly.
8. At last I used opencv library to view all test frame with anomaly labeled frames.
9. I shared output video in gamil.