HW4

學號:110062209

姓名:簡晟棋

1.

結果:

由上往下為空間、時間、整體的 micro_auc

```
(smoothing: True): dataset = avenue, auc = 0.6735457960588469
(smoothing: True): dataset = avenue, auc = 0.7715548173346876
(smoothing: True): dataset = avenue, auc = 0.734144242838198
```

2.

在 dataset.py 把 time 的正常、異常出現率改為各半，並把正常排列(np.arange)標上 class 0，異常排列(np.random.permttation)標上 class 1

```
def __getitem__(self, idx):
    temproal_flag = idx % 2 == 0
    record = self.objects_list[idx]
    t_label = np.array([0])
    if self.test_stage:
        perm = np.arange(self.frame_num)
    else:
        if random.random() < 0.5:
            perm = np.arange(self.frame_num)
        else:
            perm = np.random.permutation(self.frame_num)
            t_label = np.array([1])
    obj = self.get_object(record["video_name"], record["frame"], record["object"])
```

```
    ret = {"video": record["video_name"], "frame": record["frame"], "obj": obj, "label": t_label,
        "trans_label": spatial_perm, "loc": record["loc"], "aspect_ratio": record["aspect_ratio"], "temporal": temproal_flag}
    return  ret
```

在 model.py 改成 2 class 的 classifier

```
self.classifier_1 = nn.Sequential(
    nn.Linear(1024, 512),
    nn.ReLU(),
    nn.Linear(512, 2)
)
```

在 main 中把 train 與 val 中的 shape 改成 1 * 2 個 class

anomaly score = class 1 的機率

```
temp_logits, spat_logits = net(obj)
temp_logits = temp_logits[t_flag].view(-1, 2)
spat_logits = spat_logits[~t_flag].view(-1, 9)
```

```
with torch.no_grad():
    temp_logits, spat_logits = net(obj)
    temp_logits = temp_logits.view(-1, 1, 2)
    spat_logits = spat_logits.view(-1, 9, 9)
```

```
temp_probs = F.softmax(temp_logits, -1)
scores2 = temp_probs[:,:,1].view(-1).cpu().numpy()
```

結果:

由上往下為空間、時間、整體的 micro_auc

```
(smoothing: True): dataset = avenue, auc = 0.683344125747892
(smoothing: True): dataset = avenue, auc = 0.6222677819813986
(smoothing: True): dataset = avenue, auc = 0.6706750877068484
```

3.

在 dataset.py 把 time label 標上該排列在 permutations 中的編號，正常為 0

```
def __getitem__(self, idx):
    temproal_flag = idx % 2 == 0
    record = self.objects_list[idx]
    t_label = np.array([0])
    t_stand = tuple(i for i in range(self.frame_num))
    t_per = permutations(t_stand)
    if self.test_stage:
        perm = np.arange(self.frame_num)
    else:
        perm = np.random.permutation(self.frame_num)
        t_label = np.array([list(t_per).index(tuple(i for i in perm))])
    obj = self.get_object(record["video_name"], record["frame"], record["object"])
```

在 model.py 改成 120 class 的 classifier

```
self.classifier_1 = nn.Sequential(
    nn.Linear(1024, 512),
    nn.ReLU(),
    nn.Linear(512, 120)
)
```

在 main 中把 train 與 val 中的 shape 改成 1 * 120 個 class

anomaly score = 1 - class 0(正常)的機率

```
temp_logits, spat_logits = net(obj)
temp_logits = temp_logits[t_flag].view(-1, 120)
spat_logits = spat_logits[~t_flag].view(-1, 9)
```

```
with torch.no_grad():
    temp_logits, spat_logits = net(obj)
    temp_logits = temp_logits.view(-1, 1, 120)
    spat_logits = spat_logits.view(-1, 9, 9)
```

```
temp_probs = F.softmax(temp_logits, -1)
scores2 = 1 - temp_probs[:,:,0].view(-1).cpu().numpy()
```

結果:

由上往下為空間、時間、整體的 micro_auc

```
(smoothing: True): dataset = avenue, auc = 0.6866328553624255
(smoothing: True): dataset = avenue, auc = 0.5874238802130689
(smoothing: True): dataset = avenue, auc = 0.670117134658287
```

4.

micro_auc 在時間方面的表現為 method 1(原版)遠高於 method 2(2 classifier)與 method 3(permutation classfier)，且 method 2 只比 method 3 略高一點，原因可能為 method 2 構造太簡單，而 method 3 則是 構造複雜，只用 20 個 epoch 無法 train 出好的效果。在空間方面由於沒有改動因此 3 個方法表現都差不多，導致整體方面的表現與時間方面的表現為正相關。