# CSCE689 Project

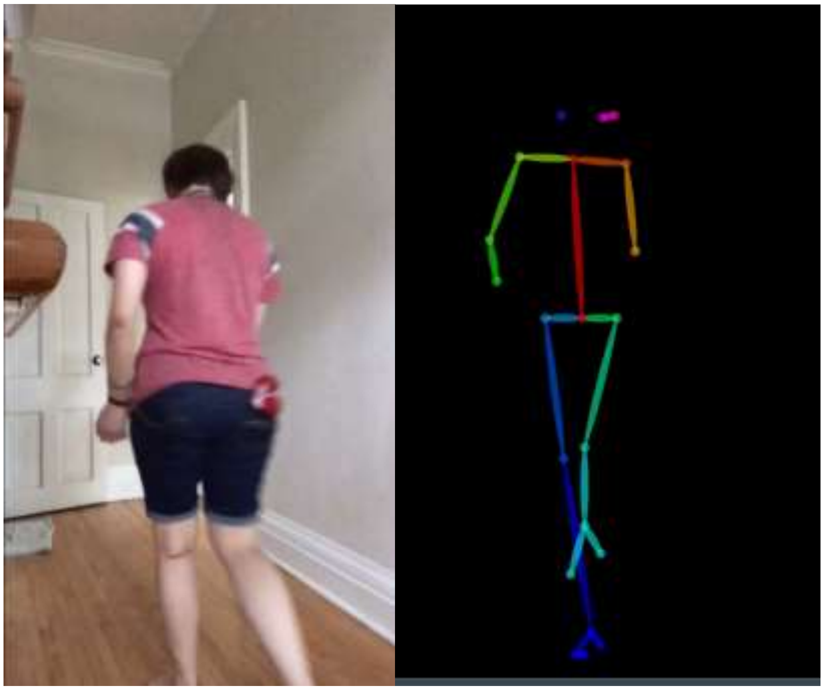
**Name: Pei Chen    UIN: 130005135**

## Research Topic:

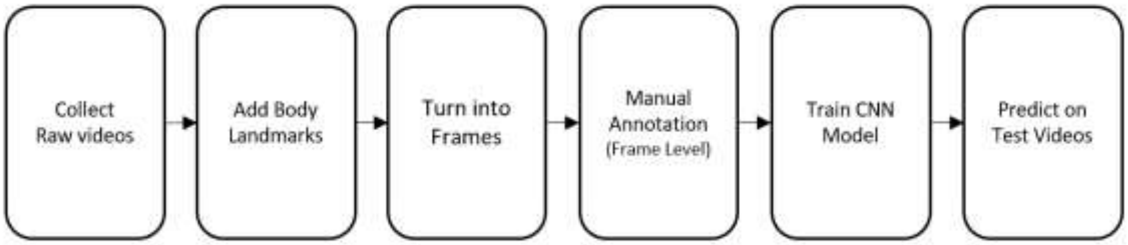
Detecting foot pacing in videos. Many people will pace when they are stressed. This acts as a pacifier, as all repetitive behaviors do. I will detect such behaviors in the videos.

## Dataset:

I am required to detect 3 types foot movement in videos. Because I still do not find videos about Foot Withdrawing and Foot Turning Away, I will start from a Foot Pacing video collected from YouTube. The video has 35 seconds and 1032 frames in total. I use the first 700 frames as the training data and the rest as test data.

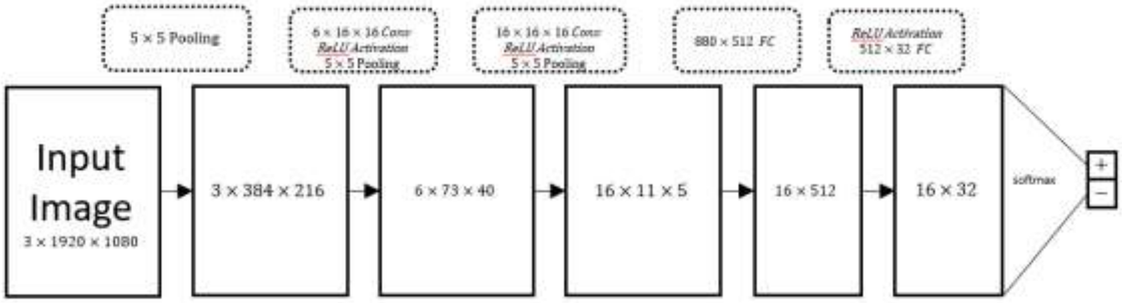


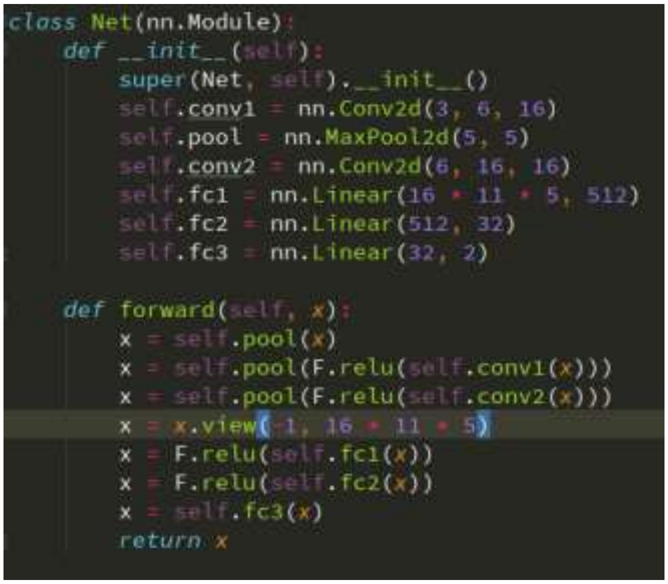
# Procedures:



1. Collect raw videos from YouTube;
2. Use tools mark “Body Landmarks” in the videos and **use the landmark video only**;
3. Then transform the videos into frames, each frame is an image;
4. After that, annotate each frame as Positive (with pacing) or Negative (without pacing);
5. Split the frames into training data and testing data and train the model.
6. Predict the test video with the CNN model to classify the frames.

# Architecture:





```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(3, 6, 16)
        self.pool = nn.MaxPool2d(5, 5)
        self.conv2 = nn.Conv2d(6, 16, 16)
        self.fc1 = nn.Linear(16 * 11 * 5, 512)
        self.fc2 = nn.Linear(512, 32)
        self.fc3 = nn.Linear(32, 2)

    def forward(self, x):
        x = self.pool(x)
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = x.view(-1, 16 * 11 * 5)
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return x
```

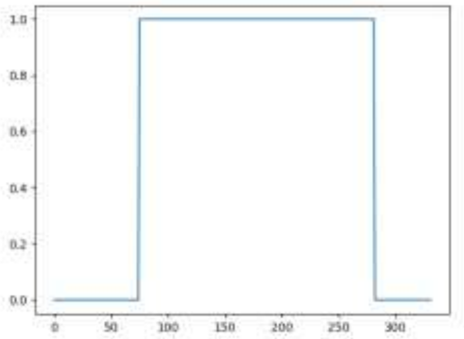
The main architecture is a 2-layer CNN. One trick I use is to first apply max-pooling in the data, because the images with body landmarks do not need too many details to distinguish the pacing behaviors. Another trick is that I use a 2-layer FNN for the final classification.

# Input:

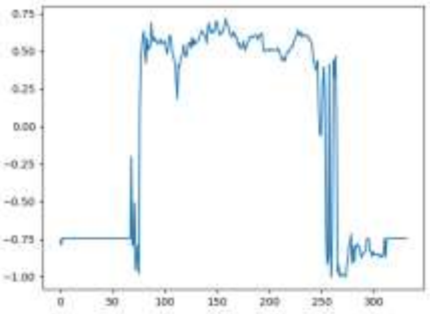
Images (Frames) with shape is (3,1920,1080). We have 700 samples in the training data and 332 samples in the test data.

# Output:

Scores at frame level, in ‘.png’ and ‘.json’ format.



Gold labels



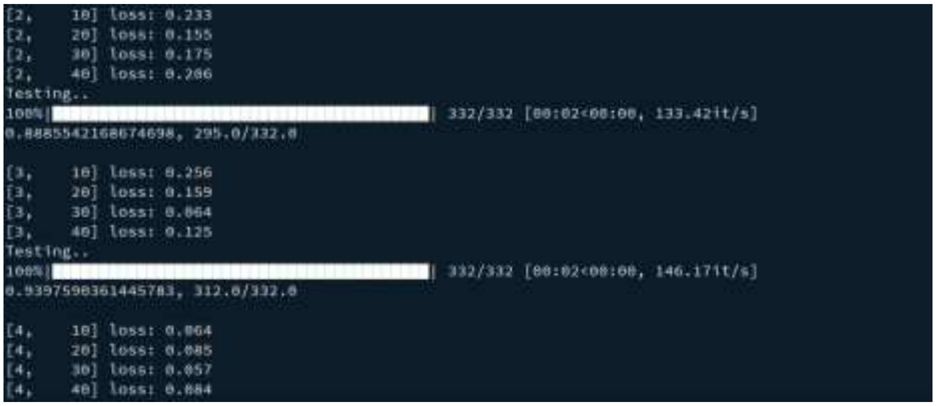
Predict Scores

# Hyperparameters:

Batch_size: 16; Optimization: SGD; Learning Rate: 0.001; Max_epoch: 30

# Training and Testing Performance:

Test Accuracy is 93.98%



```
[2,    10] loss: 0.233
[2,    20] loss: 0.155
[2,    30] loss: 0.175
[2,    40] loss: 0.206
Testing..
100%|██████████████████████████████████████████| 332/332 [00:02<00:00, 133.42it/s]
0.8885542168674698, 295.0/332.0

[3,    10] loss: 0.256
[3,    20] loss: 0.159
[3,    30] loss: 0.064
[3,    40] loss: 0.125
Testing..
100%|██████████████████████████████████████████| 332/332 [00:02<00:00, 146.17it/s]
0.9397590361445783, 312.0/332.0

[4,    10] loss: 0.064
[4,    20] loss: 0.085
[4,    30] loss: 0.057
[4,    40] loss: 0.084
```

# Next Steps:

1. Collect more videos to enlarge the datasets.
2. Improve the model.