Recognizing Human Gait Signatures with GOTCHA

The Project

1. Extract consecutive frames from the videos
2. Use Open Pose to detect the skeleton of the subject in the frame
3. Store the coordinates of consecutive frame feature points in an array (information about the subject movement variations in time).
4. Create a NN that use as input the pattern array and give us the user id as output:
5. Use a lot of pattern array samples to train the network;
6. Try to reach the highest accuracy chosing the best loss function.
7. Classify if a user is indoor or outdoor

Libraries used

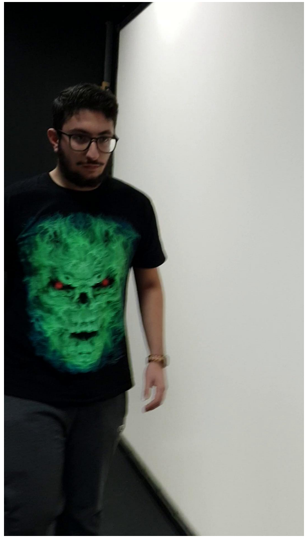
scikit learn

OpenCV

pandas

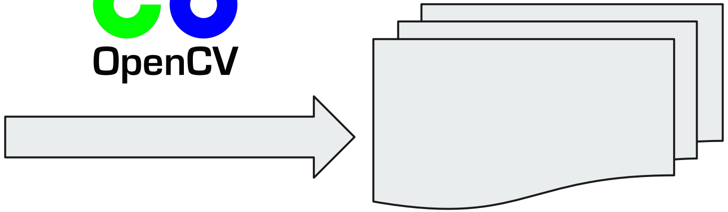
matplotlib

1) Extract consecutive frames from the videos



1920x1080x60
.mp4

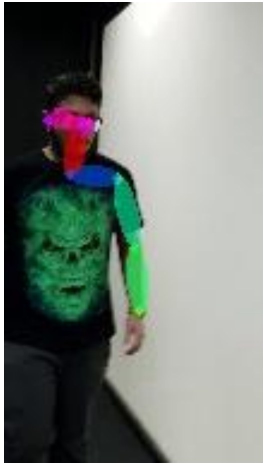
OpenCV



192 x 108
140 - 200 .jpegs

- only used folders 1 to 6 of the dataset
- started extracting after one third of the video
- resized and rotated all frames to reduce dimensions

2) Use Open Pose to detect the skeleton of the subject in the frame



```
32 53 0.84647
25 72 0.72675
0 72 0.51692
NaN
NaN
47 73 0.70893
52 101 0.79608
54 127 0.76814
8 136 0.37674
10 179 0.37429
NaN
35 137 0.46747
28 181 0.45214
NaN
27 48 0.86218
35 48 0.91030
18 48 0.83072
40 49 0.38128
```

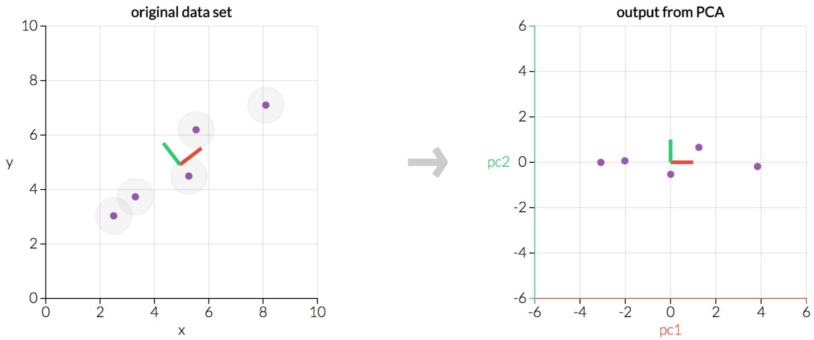
- **OpenPose** isn't robust to rotation
- A lot of joints missed, partly because of the way videos are shot (people in the BG, black t-Shirts in front of black BG)
- Extracted the XY coordinates of each joint over time
- Interpolation of missing data

3) Store the coordinates of consecutive frame feature points in an array (information about the subject movement variations in time).

	percent_missing
neckX	0.176255
neckY	0.176255
noseX	0.410344
noseY	0.410344
left_shoulderX	0.692628
...	...
right_kneeX	55.524497
right_ankleY	79.546143
right_ankleX	79.546143
left_ankleX	81.109030
left_ankleY	81.109030

36 rows × 1 columns

- ➢ removing columns with too much missing data as well as ears and eyes
- ➢ defined additional features like the vertical differences between both shoulders, hips and elbows.
- ➢ Scaled everything with Scikit StandardScaler



$$\mathbf{f} = [\mathbf{j}_1, \mathbf{j}_2, \mathbf{j}_3, \mathbf{j}_4, \mathbf{j}_5, \mathbf{j}_6, \mathbf{j}_7, \mathbf{j}_8, \mathbf{j}_9, \mathbf{j}_{10}, \mathbf{j}_{11}] \tag{1}$$

$$\mathbf{j}_i = \frac{J_i}{s} + T, \qquad 1 \leq i \leq 11 \tag{2}$$

$$s = \frac{\|J_4 - J_2\|}{h} \tag{3}$$

- ➢ Performed **PCA** to reduce redundant information
- ➢ Separated the Dataframe in packages of 20 consecutive frames for each video.
- ➢ normalized all coordinates for scale by making the hip the pivot and scaling all samples according to one reference scale which is the distance from hip to neck.

Start

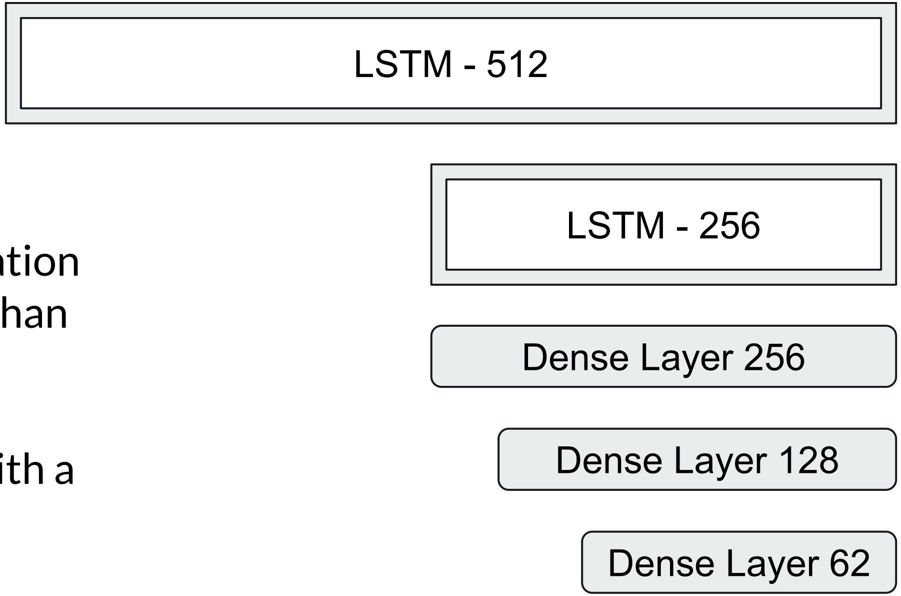
	ID	VideoNr	FrameNr	Folder	noseX	noseY	neckX	neckY	right_shoulderX	right_shoulderY	right_elbowX	right_elbowY	right_wristX	right_wristY
0	001	1_001	0000	1	32.0	53.0	25.0	72.0	0.0	72.0	12.0	134.0	8.0	170.0
1	001	1_001	0001	1	31.0	52.0	25.0	72.0	0.0	72.0	12.0	134.0	8.0	170.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
72529	062	6_062	0196	6	52.0	80.0	48.0	103.0	26.0	104.0	18.0	141.0	24.0	171.0
72530	062	6_062	0197	6	52.0	80.0	49.0	103.0	26.0	105.0	18.0	141.0	25.0	172.0
72531	062	6_062	0198	6	52.0	80.0	48.0	104.0	25.0	105.0	18.0	141.0	24.0	173.0
72532	062	6_062	0199	6	52.0	80.0	48.0	104.0	24.0	106.0	18.0	143.0	25.0	175.0
72535	062	6_062	0200	6	52.0	80.0	48.0	105.0	25.0	106.0	18.0	143.0	24.0	174.0

72622 rows × 28 columns

End

	ID	Folder	Videoname	Gait
0	01	1	1_001	[[6.145159768783847, -3.5423749813550796, -3.3708293723367886, 0.23128835707276635, -1.159242353...
1	01	1	1_001	[[1.9886155738568825, -4.921972112441757, -2.0944874182306568, 0.8068518798514708, -2.7840735532...
2	01	1	1_001	[[1.1275399489969578, -4.707025450859268, 0.1293137546816684, 0.7045429655107176, -4.02894030603...
3	01	1	1_001	[[0.9547622608992645, -4.496748724485409, 2.5327771286190743, -0.19808121615317503, -3.549206571...
4	01	1	1_001	[[3.0437848608634366, -3.802569259289150 7, -1.193002683227392, 2.04956855385584, -0.821966242930...
...	...	...	...	...

4) Create a NN that can predict the users ID from the Gait Pattern Array



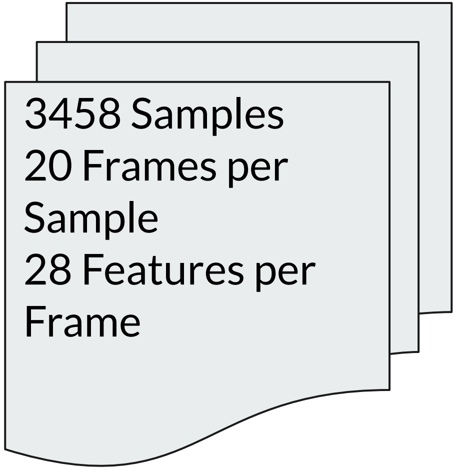
CuDNNLSTM supported by nvidia optimization for parallel computation is 15 times faster than the default keras LSTM.

Stochastic Gradient Descent Optimizer with a learning-rate of 1e-4, decay towards 1e-6, momentum of 0.9 and nesterov.

5) Use a lot of pattern array samples to train the network;

Balanced class weights labels due to uneven number of samples for all classes

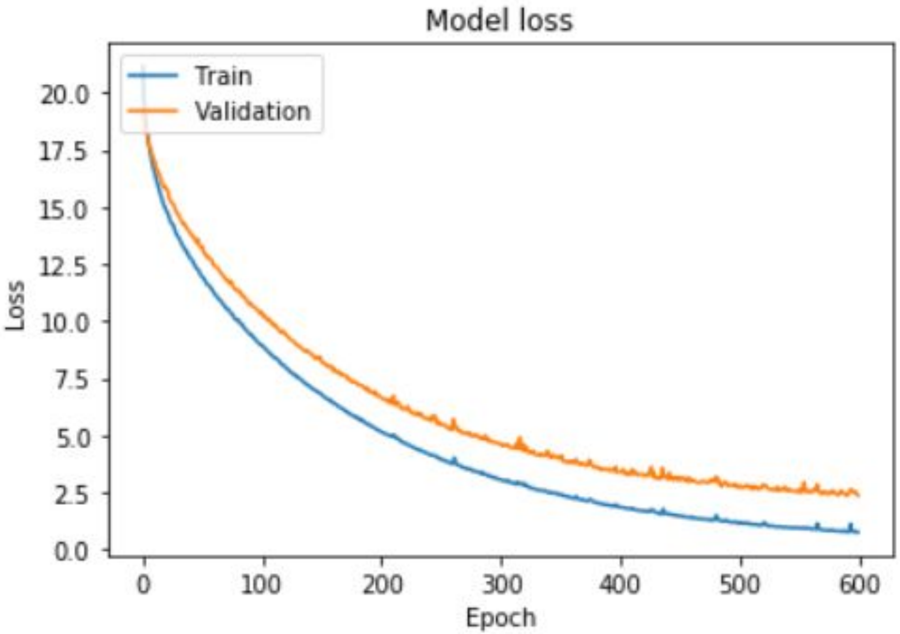
Trained with 600 Epochs and a batch size of 16



6) Try to reach the highest accuracy chosing the best loss function

Categorical Crossentropy was used for the multi-class classification loss

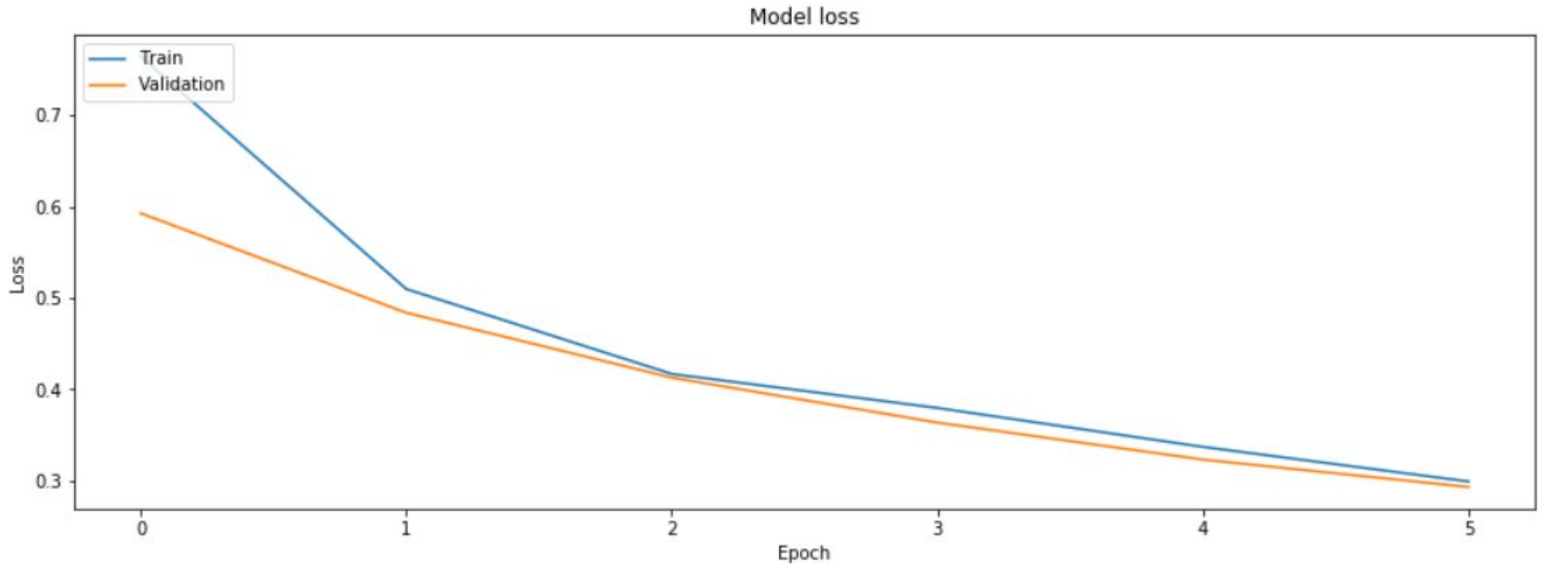
	precision	recall	f1-score	support
1	1.00	0.50	0.67	8
2	0.60	1.00	0.75	3
3	1.00	1.00	1.00	2
4	0.57	0.57	0.57	7
5	0.88	0.58	0.70	12
6	0.64	0.70	0.67	10
7	0.62	0.71	0.67	7
8	0.57	0.80	0.67	5
9	0.62	0.67	0.64	12
10	0.64	0.88	0.74	8
micro avg	0.64	0.64	0.64	450
macro avg	0.60	0.61	0.58	450
weighted avg	0.66	0.64	0.63	450



Over all epochs the Network achieved on average 66% weighted accuracy.

The highest peak in accuracy achieved was 72.59%

7) Classify if a user is indoor or outdoor



1) Extracted one frame from every video
2) Calculate a distribution histogram of brightness values (color if necessary)
3) Train a small network consisting of 2 Dense Layers with 3 and 4 neurons.

98.64% Accuracy after only 6 epochs, also works when shown a completely new frame from the same dataset.

Thank you

References:

[1] Li, S., Cui, L., Zhu, C., Li, B., Zhao, N., & Zhu, T. (2016). Emotion recognition using Kinect motion capture data of human gaits. PeerJ, 4, e2364.

[2] Gaglio, S., Re, G. L., & Morana, M. (2014). Human activity recognition process using 3-D posture data. IEEE Transactions on Human-Machine Systems, 45(5), 586-597.