Deep Learning-Based Gait Recognition Using Smartphones in the Wild [1]

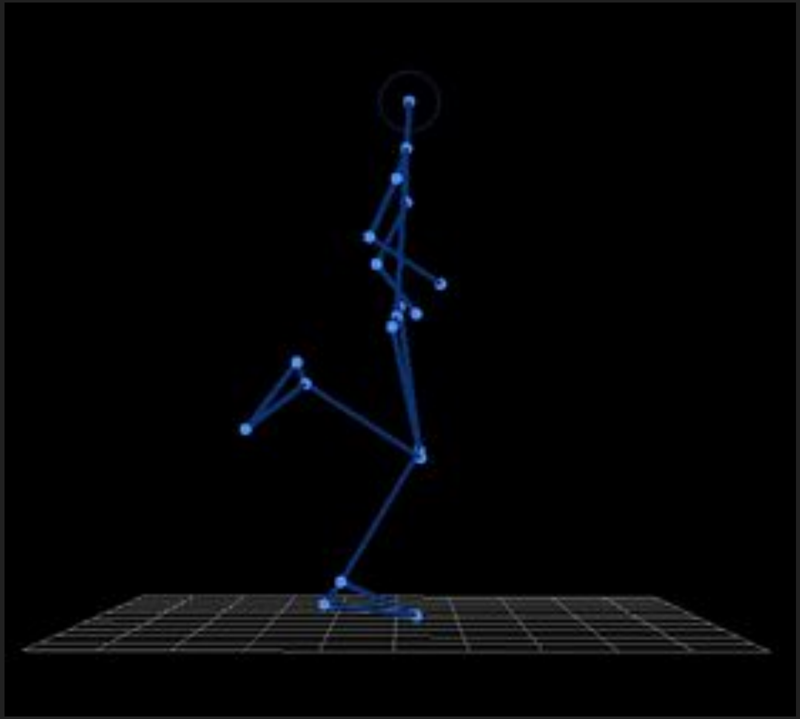
[1]Aniruddha Anand Damle
[1]Prakriti Biswas
[1]Aditya Shrikant Kaduskar

[1]School of Electrical, Computing and Energy Engineering
Ira A. Fulton Schools of Engineering
Arizona State University
April 19th, 2022

[1]Qin Zou , Senior Member, IEEE, Yanling Wang, Qian Wang , Senior Member, IEEE, Yi Zhao, and Qingquan Li

Introduction

- Problem Statement: Using deep learning methods to identify and authenticate a person by their gait data provided by an application on their smartphones.
- The project comprises three subparts: Gait data extraction, Gait Identification, and Authentication.
- The motive of the paper is to propose an effective and seamless combination of DCNN and DRNN for robust inertial gait feature representation.



Datasets

WhuGAIT Dataset: 3-axis accelerometer and 3 axis gyroscope data, collected at a sampling rate of 50 Hz using an Android application. All data is pre-processed using Gait Segmentation algorithm. Train/Test split of 90%-10%. Length of each sample is 128. Comprises of the following datasets:

- Dataset 1 (Identification): 118 subjects. 33,104 Training samples, 3740 Testing samples.
- Dataset 2 (Identification): 44,339 samples are used for training, and the rest 4,936 for testing.
- Dataset 3 (Identification): 26,283 samples are used for training, and the rest 2,991 for testing.
- Dataset 4 (Identification): 35,373 samples are used for training, and the rest 3,941 for test.
- Dataset 5 (Authentication): There are 66,542 samples and 7,600 samples for training and test, respectively. Horizontally aligned data.
- Dataset 6 (Authentication): Contains same data as dataset 5. Vertically aligned data.
- Dataset 7 (Segmentation): 577 samples from 10 subjects, with data shape of 6×1024. 519 of them were used for training and 58 were used for testing.
- Dataset 8 (Segmentation): 1,354 samples from 118 subjects, with data shape of 6×1024. 1022 samples from 20 subjects as training data and 332 samples from other 98 subjects for testing.

Data Extraction

- Smartphones to collect the inertial data
- Walking sessions and non-walking sessions
- Partitioning problem as a time-series segmentation problem
- Inspired by U-Net
- Semantic segmentation algorithm with a one-dimensional DCNN

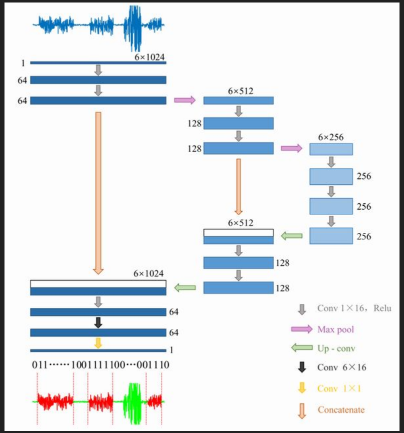


Fig. 1: Data Extraction Network Architecture

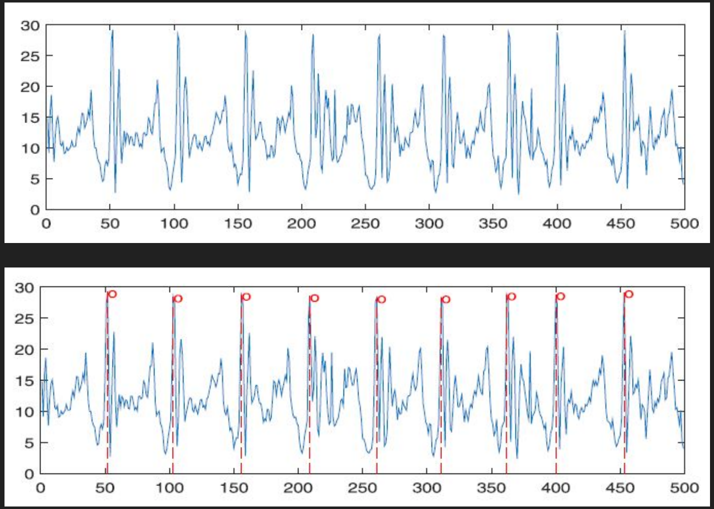
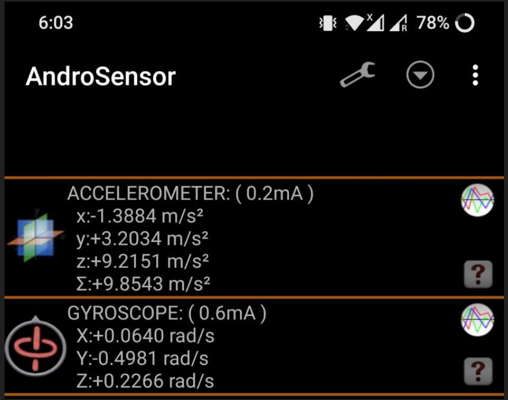


Fig. 2: An example of step segmentation. (a) The gait curve ACCo, calculated on the three components as shown in Figure 2. (b) The segmentation results, where the red dots denote the local maximums and divide the steps.

Data Extraction Implementation

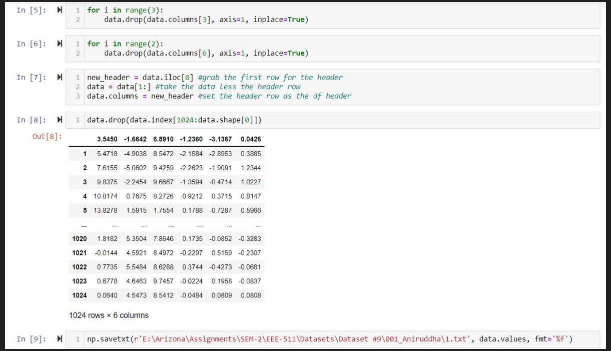

6:03 78%

AndroSensor

ACCELEROMETER: (0.2mA)
x:-1.3884 m/s²
y:+3.2034 m/s²
z:+9.2151 m/s²
Σ:+9.8543 m/s²

GYROSCOPE: (0.6mA)
X:+0.0640 rad/s
Y:-0.4981 rad/s
Z:+0.2266 rad/s

Fig. 3: AndroSensor Android Application

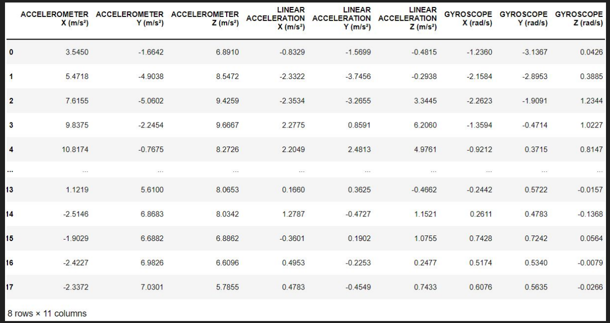

```
In [5]:  for i in range(3):
             data.drop(data.columns[3], axis=1, inplace=True)

In [6]:  for i in range(2):
             data.drop(data.columns[6], axis=1, inplace=True)

In [7]:  new_header = data.iloc[0] #grab the first row for the header
         data = data[1:] #take the data less the header row
         data.columns = new_header #set the header row as the df header

In [8]:  data.drop(data.index[1024:data.shape[0]])
Out[8]:
```

	3.5450	-1.6642	6.8910	-1.2360	-3.1367	0.0426
1	5.4718	-4.9038	8.5472	-2.1584	-2.8953	0.3885
2	7.6155	-5.0602	9.4259	-2.2623	-1.9091	1.2344
3	9.8375	-2.2454	9.6667	-1.3594	-0.4714	1.0227
4	10.8174	-0.7675	8.2726	-0.9212	0.3715	0.8147
5	13.8278	1.5915	1.7554	0.1788	-0.7287	0.5966
...	...	...	...	...	...	...
1020	1.8182	5.3504	7.8646	0.1735	-0.0852	-0.3283
1021	-0.0144	4.5921	8.4972	-0.2297	0.5159	-0.2307
1022	0.7735	5.5484	8.6288	0.3744	-0.4273	-0.0681
1023	0.6778	4.6463	9.7457	-0.0224	0.1958	-0.0837
1024	0.0640	4.5473	8.5412	-0.0484	0.0809	0.0808

1024 rows × 6 columns

```
In [9]:  np.savetxt(r'E:\Arizona\Assignments\SEM-2\EEE-511\Datasets\Dataset #9\001_Aniruddha\1.txt', data.values, fmt='%f')
```

Fig. 4: Data Formatting



	ACCELEROMETER X (m/s²)	ACCELEROMETER Y (m/s²)	ACCELEROMETER Z (m/s²)	LINEAR ACCELERATION X (m/s²)	LINEAR ACCELERATION Y (m/s²)	LINEAR ACCELERATION Z (m/s²)	GYROSCOPE X (rad/s)	GYROSCOPE Y (rad/s)	GYROSCOPE Z (rad/s)
0	3.5450	-1.6642	6.8910	-0.8329	-1.5699	-0.4815	-1.2360	-3.1367	0.0426
1	5.4718	-4.9038	8.5472	-2.3322	-3.7456	-0.2938	-2.1584	-2.8953	0.3885
2	7.6155	-5.0602	9.4259	-2.3534	-3.2655	3.3445	-2.2623	-1.9091	1.2344
3	9.8375	-2.2454	9.6667	2.2775	0.8591	6.2060	-1.3594	-0.4714	1.0227
4	10.8174	-0.7675	8.2726	2.2049	2.4813	4.9761	-0.9212	0.3715	0.8147
...	...	...	...	...	...	...	...	...	...
13	1.1219	5.6100	8.0653	0.1660	0.3625	-0.4662	-0.2442	0.5722	-0.0157
14	-2.5146	6.8683	8.0342	1.2787	-0.4727	1.1521	0.2611	0.4783	-0.1368
15	-1.9029	6.6882	6.8862	-0.3601	0.1902	1.0755	0.7428	0.7242	0.0564
16	-2.4227	6.9826	6.6096	0.4953	-0.2253	0.2477	0.5174	0.5340	-0.0079
17	-2.3372	7.0301	5.7855	0.4783	-0.4549	0.7433	0.6076	0.5635	-0.0266

8 rows × 11 columns

Fig. 5: Data Collected from Smartphone

Learning Rate: 0.0001
Epochs: 150

- **Dataset #7**: 577 samples of 10 subjects, with data shaped as 6 × 1,024. Among these samples, 519 are used for training and 58 are for test. Both the training and test samples are from the 10 subjects.
- **Dataset #8**: 1,354 samples of 118 subjects, with data shaped as 6×1,024. Among these data, 1,022 samples from 20 subjects are used for training, and 332 samples from the other 98 subjects are used for test.

Data Extraction Results

- ‘1’ as the walking data, and ‘0’ as the non-walking data. The labels are manually annotated.
- Dataset #7: The training data and test data have no overlap but are all from the same 10 subjects, the proposed method achieves an accuracy of 90.22%, which shows the effectiveness of the proposed method in separating walking data from non-walking data.
- Dataset #8: The training data and the test data are from different subjects, an accuracy of 85.57% is obtained, which indicates that the proposed method has high generalization ability.

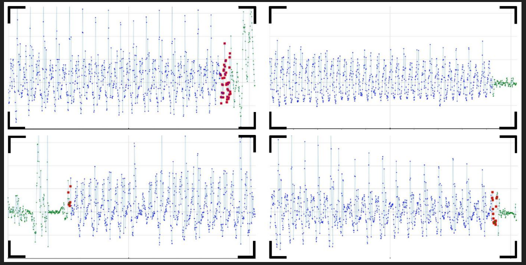


Fig. 6: Four examples of walking data extraction using the proposed method. Note that, the blue points denote the walking data, green points denote the non-walking data, and the red denotes the false classified.

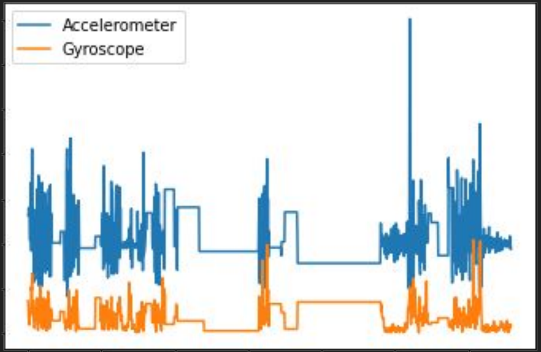


Fig. 7: Accelerometer and Gyroscope data extracted from one of our peer’s data

Identification

$$\mathbf{x} = (x_1, x_2, \ldots, x_T),$$
$$\text{w.r.t., } x_t = (acc_x^t, acc_y^t, acc_z^t, gyr_x^t, gyr_y^t, gyr_z^t)^\top$$

Eq. 1: Input Inertial Gait curve

$$\mathcal{O} = (o_1, o_2, \ldots, o_n)$$

Eq. 2: Output vector

$$s = \underset{s_i}{arg\max}\{o_i \mid 1 \leq i \leq n\}$$

Eq. 3: Identity of Input Data

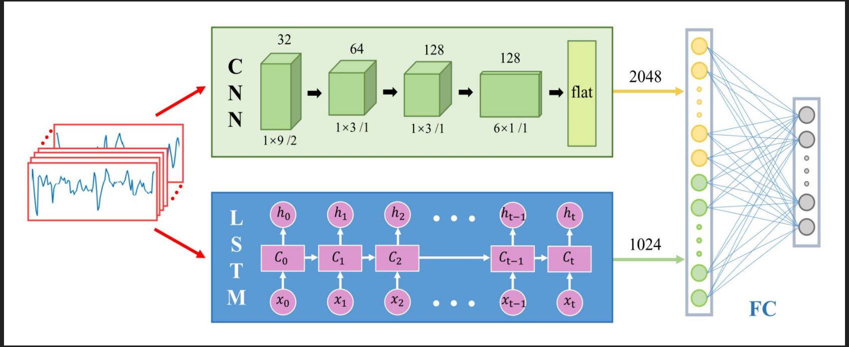

CNN
32
64
128
128
1×9 /2
1×3 /1
1×3 /1
6×1 /1
flat
2048
LSTM
h_0 h_1 h_2 h_{t-1} h_t
C_0 C_1 C_2 C_{t-1} C_t
x_0 x_1 x_2 x_{t-1} x_t
1024
FC

Fig. 8: Network Architecture for Gait Identification.

Identification Implementation

LSTM Network: Each hidden layer in the LSTM has N = 1024 hidden nodes, the learning rate is set to 0.0025, and the number of epochs for training is 200.

$$\mathbf{h}_t^l = \sigma(W_{xh}^l x_t + \mathbf{h}_{t-1}^l W_{hh}^{tl} + \mathbf{h}_t^{l-1} W_{hh}^{ll} + \mathbf{b}_h^l)$$

Eq. 4: State for each layer at time 't'

$$\begin{aligned}
\mathbf{i}_t &= \sigma_i(W_{xi} x_t + W_{hi}^t \mathbf{h}_{t-1}^l + W_{hi}^l \mathbf{h}_t^{l-1} + W_{ci} \mathbf{c}_{t-1} + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma_f(W_{xf} x_t + W_{hf}^t \mathbf{h}_{t-1}^l + W_{hf}^l \mathbf{h}_t^{l-1} + W_{cf} \mathbf{c}_{t-1} + \mathbf{b}_f) \\
\mathbf{c}_t &= \mathbf{f}_t + \mathbf{c}_{t-1} \sigma_i(W_{xc} x_t + W_{hc}^t \mathbf{h}_{t-1}^l + W_{hc}^l \mathbf{h}_t^{l-1} + \mathbf{b}_c) \\
\mathbf{o}_t &= \sigma_o(W_{xo} x_t + W_{ho}^t \mathbf{h}_{t-1}^l + W_{ho}^l \mathbf{h}_t^{l-1} + W_{co} \mathbf{c}_{t-1} + \mathbf{b}_o) \\
\mathbf{h}_t^l &= \mathbf{o}_t \sigma_h(\mathbf{c}_t)
\end{aligned}$$

Eq. 5: Updating state equation

$$\begin{aligned}
feat_{lstm} &= \mathbf{h}_T^L \\
&= (f_1, f_2, \ldots, f_N)
\end{aligned}$$

Eq. 6: LSTM Output features

CNN Network: Six-axis accelerometer / gyroscope data shaped as 6 × 128. The learning rate is 0.0025, and the number of epochs for training is 200.

Layer Name	Kernel Size	Number of features	Stride	Feature Map
Conv1_1	1 x 9	32	2	6 x 64 x 32
pool1	1 x 2	/	2	6 x 32 x 32
conv2_1	1 x 3	64	1	6 x 32 x 64
conv2_2	1 x 3	128	1	6 x 32 x 128
pool2	1 x 2	/	2	6 x 16 x 128
conv3_1	6 x 1	128	1	1 x 16 x 128

Table 1: Details of the CNN structure

Identification Implementation (Continued)

Fully Connected Layer: Concatenation of the features extracted by LSTM and CNN, i.e., $feat_{full} = feat_{LSTM} + feat_{CNN}$

$$\mathbf{o} = \mathbf{Softmax}(feat_{full} * W_o + \mathbf{b}_o)$$

Eq. 7: Classification Output

$$\mathbf{o}' = (o'_1, o'_2, \ldots, o'_n)$$

Where,

$$o'_i = \begin{cases} 1 & \mathbf{x} \in s_i, \\ 0 & other. \end{cases}$$

Eq. 8: Output Class label

$$\mathcal{L}(\mathbf{o}, \mathbf{o}') = \sum_{i}^{n} o'_i \ln o_i + (1 - o'_i) \ln(1 - o_i)$$

Eq. 9: Cross-Entropy Loss Function

Identification Results

Classification Methods	Dataset #1	Dataset #2
CNN	92.54%	95.10%
CNN+LSTM	91.42%	95.67%
CNN+LSTM$_{fix}$	**93.43%**	**98.43%**

Table 2: Classification Performance Of Deep Learning-Based Methods

- The two Dataset #1, Dataset #2, are used to evaluate the following deep learning-based methods: CNN, CNN+LSTM, and CNN+LSTM$_{fx}$.
- The classification results are shown in Table. All the methods achieve greater than 90% accuracy on both datasets. Dataset #1 contains 118 subjects, and Dataset #2 consists of the gait data of 20 subjects.
- CNN - Only CNN network is trained.
- CNN + LSTM - Both networks are trained from scratch in parallel and features are fed to fully connected layer.
- CNN + LSTM$_{fix}$ - CNN and fully connected layer are trained from scratch, but LSTM weights are fixed.

Authentication

- Two sequences of gait data x_a and x_b:

$$\mathbf{x_a} = (x_{a,1}, x_{a,2}, \ldots, x_{a,T})$$
$$\mathbf{x_b} = (x_{b,1}, x_{b,2}, \ldots, x_{b,T})$$

- Authentication is formulated as a binary classification problem
- The output of the network is set as two dimensions. We use ‘True’ and ‘False’ to denote that the input data are from the same subject and different subjects, respectively

Authentication Implementation

- CNN is a feature extractor to map input inertial signals into lower-dimensional abstractions
- CNN trained on 98 subjects and tested on 20 subjects that have no overlap.
- CNN is fixed as feature extractor
- I/p gait signal is 6x128, O/p of CNN is 1x16x128
- rearrange the CNN features into 16 × 256 for a pair of inputs
- Features fed into double-layer LSTM for training and prediction

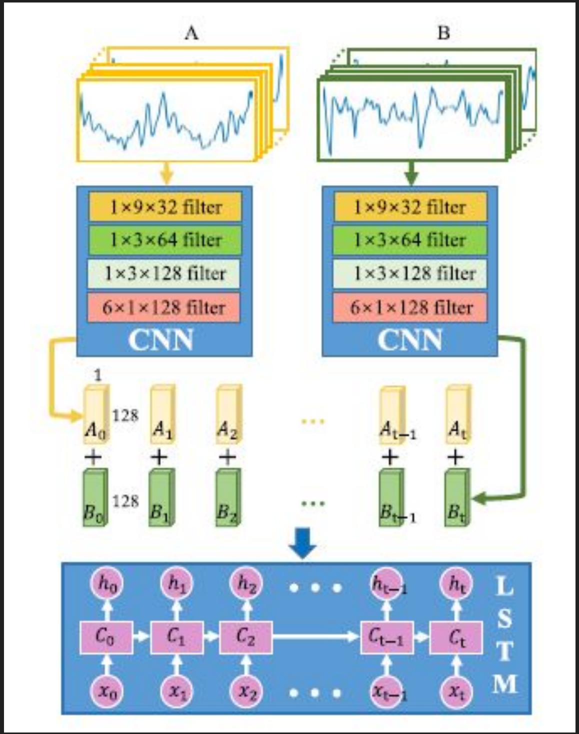

A
B
1×9×32 filter
1×3×64 filter
1×3×128 filter
6×1×128 filter
CNN
1
128
A_0 A_1 A_2 A_{t-1} A_t
B_0 B_1 B_2 B_{t-1} B_t
h_0 h_1 h_2 h_{t-1} h_t
C_0 C_1 C_2 C_{t-1} C_t
x_0 x_1 x_2 x_{t-1} x_t
LSTM

Fig. 9: The network architecture for gait authentication

Authentication Results

- On Implementing the CNN and CNNfix+LSTM architectures on dataset 5 and 6, we have received the adjacent performance table (testing accuracies)
- According to these results, CNNfix+LSTM performs more effectively on the vertical dataset (Dataset #6).

Authentication Methods	Dataset #5 (Horizontal)	Dataset #6 (Vertical)
CNN	78.3%	86.96%
CNN_{fix}+LSTM	85.48%	93.7%

Table 3: Authentication performance based on training accuracy of models

Conclusion

- For robust inertial gait feature representation, a hybrid technique was presented that smoothly combined the DCNN and DRNN.
- The smartphones were used in unrestricted conditions during gait data collection, and no information regarding when, where, or how the user walks was available.
- The suggested hybrid network outperformed the standalone networks by a wide margin.
- We got the opportunity to dive deep into machine learning frameworks such as Tensorflow, and development environments such as Jupyter Notebook and Google Colab.
- We were able to better understand many ideas related to the CNN and LSTM models.
- We discovered the many hyperparameters that have a direct impact on our model's accuracy.

System Specification & References

Systems Used:

- HP Omen 15: Identification
  - CPU: Intel i7-7700-HQ 7th Gen 3.02GHz Quad core
  - GPU: Nvidia GEFORCE GTX 1050ti 4GB
  - Ram: 16GB DDR4
- Dell G3: Authentication
  - CPU: Intel i7-8750-H8th Gen 2.2GHz Quad core
  - GPU: Nvidia GEFORCE GTX 1050ti 4GB
  - Ram: 16GB DDR4
- HP 15 bs180: Data Extraction
  - CPU: Intel i5-8250U 8th Gen 1.8GHz Quad core
  - GPU: AMD Radeon Adrenaline 2GB
  - RAM: 8GB DDR4

Software Used:

- Jupyter Notebook
- Google Colab
- Tensorflow-gpu v1.8.0

References:

[1] Q. Zou, Y. Wang, Q. Wang, Y. Zhao, and Q. Li, “Deep Learning-Based Gait Recognition Using Smartphones in the Wild,” *IEEE Transactions on Information Forensics and Security*, vol. 15. pp. 3197–3212, 2020 [Online]. Available: http://dx.doi.org/10.1109/tifs.2020.2985628

Questions?

Thank You!