# Deep Learning-Based Gait Recognition Using Smartphones in the Wild - Report*

*EEE 511: Artificial Neural Computation


Aniruddha Anand Damle
*ECEE*
*Arizona State University*
Tempe, USA
adamle5@asu.edu
ASU ID: 1222585013

Prakriti Biswas
*ECEE*
*Arizona State University*
Tempe, USA
pbiswa12@asu.edu
ASU ID: 1222851266

Aditya Shrikant Kaduskar
*ECEE*
*Arizona State University*
Tempe, USA
akaduska@asu.edu
ASU ID: 1222545896



***Abstract*—This report is a discussion of our class project where we replicate the research paper 'Deep Learning-Based Gait Recognition Using Smartphones in the Wild' by Qin Zhou et al., 2020 [1]. We attempt to replicate the models used in the aforementioned paper for the purpose of gait identification and authentication. As a first step of the purpose, we extract gait features from the accelerometer and gyroscope, which are readily available in smartphones. We used an application called 'AndroSensor' from the Google Playstore. This application generates a text file or sends an email to the user with the phone's accelerometer and gyroscope data. Our classmates utilized the aforementioned software and gave us with their gait statistics. We then applied this additional dataset to the codes provided by the original paper's authors. We thoroughly examined the research and applied the methods given by the original study's authors for gait identification utilizing cellphones. We cover the implementation of the various methodologies, the experimental findings, and any issues encountered throughout the course of this project in this report.**


## I. Introduction

The authors of the paper [1] suggest a three-step technique for identifying and authenticating a person based on the way they walk: gait data extraction, identification, and authentication. The inertial data obtained by smartphones is partitioned into walking and non-walking sessions as a consequence of the extraction phase of the proposed technique, and gait characteristics are retrieved from the walking data. The findings of the feature extraction process are used to build the person identification and authentication models. Gait is harder to disguise when compared to other biometrics, although it has the virtue of being inconspicuous. Gait dynamics are frequently captured using inertial sensors such as accelerometers and gyroscopes. Gait data is easy and affordable to acquire since inertial sensors are regularly integrated into cellphones and are extensively utilized by the typical person. We investigate gait recognition in the field using cellphones in this work. The suggested technique captures inertial gait data in unconstrained conditions without knowing when, where, or how the user walks, in contrast to previous methods, which generally require a person to walk along a specific road and/or at a standard walking speed. Deep-learning algorithms are introduced to learn and model gait biometrics based on walking data in order to achieve good person identification and authentication performance.

## II. Methods

### A. Data Extraction

*1) Introduction:* We don't know when, where, or how cellphones will be used to collect inertial data in the wild. As a consequence, although gait feature extraction and person identification are only interested in walking data, the acquired data contains both walking and non-walking sessions. As a result, the continuous inertial sequence obtained by smartphones in the wild must be partitioned.

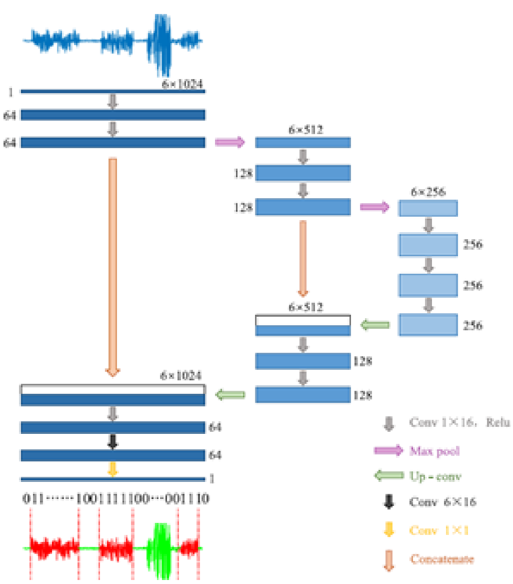


Fig. 1. Architecture of the network for gait data extraction.

TABLE I
DETAILS OF THE DATA EXTRACTION NETWORK STRUCTURE

| Layer Name | Kernel Size | Kernel Num. | Stride | Feature Map |
|---|---|---|---|---|
| conv1_1 | 1x16 | 64 | 1 | 6x1024x64 |
| conv1_2 | 1x16 | 64 | 1 | 6x1024x64 |
| pool1 | 1x2 | / | 2 | 6x512x64 |
| conv2_1 | 1x16 | 128 | 1 | 6x512x128 |
| conv2_2 | 1x16 | 128 | 1 | 6x512x128 |
| pool2 | 1x2 | / | 2 | 6x256x128 |
| conv3_1 | 1x16 | 256 | 1 | 6x256x256 |
| conv3_2 | 1x16 | 256 | 1 | 6x256x256 |
| conv3_3 | 1x16 | 256 | 1 | 6x256x256 |
| upconv1 | 1x2 | 128 | 1 | 6x512x128 |
| concat1 | / | / | 1 | 6x512x256 |
| conv4_1 | 1x16 | 128 | 1 | 6x512x128 |
| conv4_2 | 1x16 | 128 | 1 | 6x512x128 |
| upconv2 | 1x2 | 64 | 1 | 6x1024x64 |
| concat2 | / | / | / | 6x1024x128 |
| conv5_1 | 1x16 | 64 | 1 | 6x1024x64 |
| conv5_2 | 6x16 | 64 | 1 | 1x1024x64 |
| conv5_3 | 1x1 | 1 | 1 | 1X1024X1 |

We characterize the partitioning problem as a time-series segmentation problem since walking data and non-walking data are semantically distinct because inertial time series are continuous in both the spatial and time domains. We choose the accelerometer data as the foundation for partitioning without compromising generality. Because the user can move the smartphone in any direction, a single axis of acceleration or gyroscope cannot reliably reflect the oscillations in the gait curve. Meanwhile, the absolute acceleration along the perpendicular axis to the ground is the greatest of the three values. We analyze the triaxial acceleration data to generate $ACC_o$ as the foundation for gait cycle segmentation in order to eliminate the impact of the phone's orientation. ACCo is calculated by $ACC_o = \sqrt{ACC_x^2 + ACC_y^2 + ACC_z^2}$, where $ACC_x$, $ACC_y$ and $ACC_z$ denote the values of acceleration in the X, Y, and Z directions, respectively.

TABLE II
DETAIL INFORMATION OF THE GAIT-DATA EXTRACTION DATASETS

| Dataset Name | Number of Subjects | Samples for Training | Samples for Test |
|---|---|---|---|
| Dataset #7 | 10 | 519 | 58 |
| Dataset #8 | 118 | 1022 | 332 |

Two datasets are constructed for evaluation of the proposed gait-data-extraction method. Basic information of the two datasets have been shown in Table 2, and the details are given as below:

- **Dataset #7:** it contains 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:** It contains 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. For both datasets, each sample is attached with a label file, which contains 1,024 binary values, with '1' as the walking data, and '0' as the non-walking data. The labels are manually annotated.

### B. Identification

*1) Introduction:* Given an inertial gait curve X with a sampling length of T , X can be expressed as:

$$X = (x_1, x_2, ..., x_t) \tag{1}$$

where $X_t = (ACC_x, ACC_y, ACC_z, GYR_x, GYR_y, GYR_z)$, [1] where ($ACC_x$,$ACC_y$, $ACC_z$) and ($GYR_x$, $GYR_y$ $GYR_z$) denote the accelerometer and gyroscope components along the X, Y and Z axes at time t, respectively. Then, the problem is how to recognize the identity of a subject based on input data x. To formulate this problem, suppose $S = (S1, S2 \ldots, SN)$ is the classes allotted to n number of candidates, where $S_i$ is the $i^{th}$ subject. Then, the output can be represented as an n-dimensional vector,

$$O = (O_1, O_2, ..., O_n)) \tag{2}$$

where $o_i = P(s_i|x)$, i.e., the possibility that x belongs to $s_i$. Further suppose that $s$ is the identity of the input data $x$; then, it is formulated as

$$S = argmax(Si(O_i|1 \leq i \leq n) \tag{3}$$

Thus, to solve the problem of gait identification, we have to associate the maximum possibility values with the corresponding subjects. Table III specifies the datasets required for the classification portion of this project, along with the number of subjects, samples for training, and samples for testing.

TABLE III
DETAIL INFORMATION OF THE CLASSIFICATION DATASETS

| Dataset Name | Number of Subjects | Samples for Training | Samples for Test |
|---|---|---|---|
| Dataset #1 | 118 | 33104 | 3740 |
| Dataset #2 | 20 | 44339 | 4936 |
| Dataset #3 | 118 | 26283 | 2991 |
| Dataset #4 | 20 | 35373 | 3941 |

- **Dataset #1:** Gait samples were collected by dividing the gait curve into two continuous steps. The gait samples are sorted in time and split in a ratio of 9:1 corresponding to training samples to testing samples, with no overlap between the two subsets.
- **Dataset #2:** Similar to dataset #1, the gait curve is divided into two-step samples and interpolated to a length of 128. In this dataset, there is a larger amount of data as compared to Dataset #1.
- **Dataset #3:** The same 118 subjects from Dataset #1 provide the samples for Dataset #3. This dataset is different from the other as it is derived from dividing the gait curve by using a fixed time length (2.56 seconds[1]) instead of step length. While the data collecting frequency is 50Hz,

the length of each sample is also 128. In addition, a 1.28-second overlap is added to the dataset to make it larger.

- **Dataset #4:** This dataset is similar to Dataset #3 with the difference that it has no overlap between samples

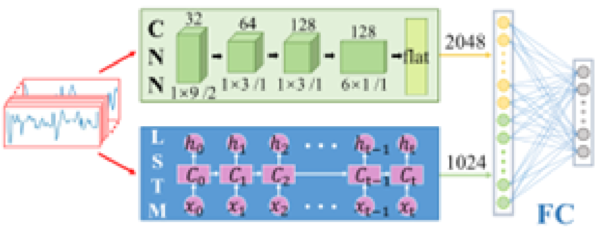


Fig. 2. Architecture for the network for Authentication

*2) Architecture:* As illustrated in Figure 2, the gait identification network consists of a CNN and an LSTM in parallel. The CNN and LSTM networks are feature extractors that obtain the corresponding features – $\text{Feat}_{\text{CNN}}$ and $\text{Feat}_{\text{LSTM}}$ The fully connected layer works as a classifier and uses the feature vector obtained by concatenating $\text{Feat}_{\text{CNN}}$ and $\text{Feat}_{\text{LSTM}}$ as the input.

### C. Authentication

*1) Introduction:* After receiving the output of the Identification segment of the project, which are the identities of all the subjects involved, the output of the Identification model is fed into the Authentication segment of the project, to authenticate their identities. This segment utilizes an architecture that is a hybrid of a fixed-parameter CNN and LSTM. Positive and negative samples account for half of the total number of samples. Each authentication sample includes a pair of data samples from two separate individuals or from the same subject. The data sample is made up of 2-step acceleration and gyroscopic data that have been interpolated in the manner indicated in Dataset #1. To make an authentication sample, the two data samples are horizontally aligned. The authentication samples in Dataset #6 are constructed in the same way as in Dataset #5, the only difference being that the two data samples from two subjects are vertically aligned.

TABLE IV
DETAIL INFORMATION OF THE AUTHENTICATION DATASETS

| Dataset Name | Number of Subjects | Samples for Training | Samples for Test |
|---|---|---|---|
| Dataset #5 | 118 | 66542 | 7600 |
| Dataset #6 | 118 | 66542 | 7600 |

*2) Architecture:* Let two sequences of gait data $x_a$ and $x_b$ be the input of the authentication network,

$$x_a = (x_{a,1}, x_{a,2}, ..., x_{a,T})$$
$$x_b = (x_{b,1}, x_{b,2}, ..., x_{b,T}) \quad (4)$$

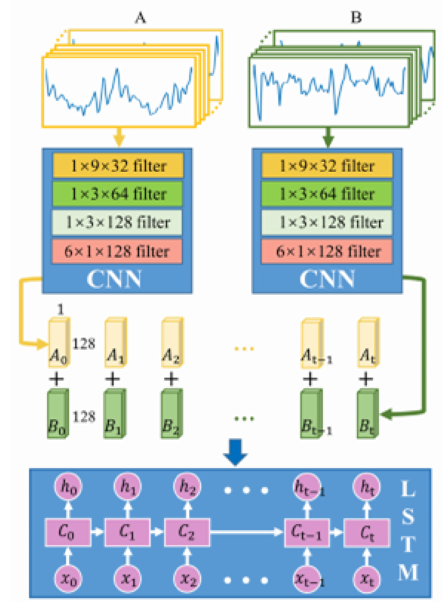


Fig. 3. Architecture for the network for Authentication

where,

$$x_t = (ACC_x^t, ACC_y^t, ACC_z^t, GYR_x^t, GYR_y^t, GYR_z^t) \quad (5)$$

And T is the length of the sequence. authentication is formulated as a binary classification problem. The output of the network is set as two dimensions. The authors of the paper [1] have used 'True' and 'False' to denote that the input data are from the same subject and different subjects, respectively. To fully utilize the advantages of the CNN and RNN, the CNN is used as a feature extractor to map the input inertial signals into lower-dimensional abstractions. The CNN portion of the network uses 98 subjects of the 118 total subjects, for training purposes. The remaining 20 subjects are used to test the CNN architecture. The 98 subjects have nothing in common with the remaining 20 subjects to prevent overfitting. The input to the CNN is the gait signal with dimensions of 6 rows and 128 columns. The 6 rows represent the x,y, and z signals of the accelerometer and the gyroscope, and the 128 columns represent the number of samples. The output of the CNN is based on the dimension requirements of the LSTM. The output of the CNN has the dimensions 1x16x128, that is, 1 layer of a 16x128 matrix. The output comprises 16 features along the time axis for the 128 samples. The CNN features are rearranged into 16 blocks of 256 features, and fed into the LSTM block for training and prediction.

## III. Implementation and Simulation

### A. Data Extraction

To collect our own dataset, we installed the "AndroSensor" android app on our smartphones. We gave this app to our friends and gathered their walking data. The dataset is being compiled, and once it is complete, it will be posted to our

GitHub account. We wrote a Python script. to convert this data to the format of the neural network's input.The accelerometer and gyroscope have a 50Hz frequency, and the data from these two sensors is captured in real time. The time stamp, the triaxial values of the acceleration sensor, and the triaxial values of the gyroscope are among the seven dimensions of the collected data.

## B. Identification

A number of network structures, including LSTM-based, CNN-based, and CNN+LSTM-based, are designed for gait classification, and their performances are compared in terms of accuracy. For LSTM-based methods, each hidden layer in the LSTM has N = 64 hidden nodes, the optimizer used is Adam [2] optimizer with the learning rate set to 0.0025, and the number of epochs for training is 200. For the hybrid method, N = 1024 is set for LSTM. For CNN-based methods, the six-axis interpolation data are used as the input, with the data shaped as 6 × 128. The classification experiments are conducted on the first four datasets introduced in Table III. When training the CNN, the learning rate is 0.0025, and the number of epochs for training is 200.

## C. Authentication

A number of network structures, including LSTM-based, CNN-based, and CNN+LSTM-based, are designed for gait classification, and their performances are compared in terms of accuracy. For LSTM-based methods, each hidden layer in the LSTM has N = 64 hidden nodes, the optimizer used is the Adam [2] optimizer with the learning rate set to 0.0025, and the number of epochs for training is 200. For the hybrid method, $N = 1024$ is set for LSTM. For CNN-based methods, the six-axis interpolation data are used as the input, with the data shaped as 6 × 128. The classification experiments are conducted on the first four datasets introduced in Section V-A. When training the CNN, the learning rate is 0.0025, and the number of epochs for training is 200.

# IV. Results

## A. Data Extraction

*1) Data Collection:* The following table shows a snippet of the data collected using the android application.

TABLE V
SNIPPET OF COLLECTED DATA

| ACCx | ACCy | ACCz | GYRx | GYRy | GYRz |
|---|---|---|---|---|---|
| 5.4718 | -4.9038 | 8.5472 | -2.1584 | -2.8953 | 0.3885 |
| 7.6155 | -5.0602 | 9.4259 | -2.2623 | -1.9091 | 1.2344 |
| 9.8375 | -2.2454 | 9.6667 | -1.3594 | -0.4714 | 1.0227 |
| 10.8174 | -0.7675 | 8.2726 | -0.9212 | 0.3715 | 0.8147 |
| 13.8278 | 1.5915 | 1.7554 | 0.1788 | -0.7287 | 0.5966 |

| | 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. 4. Snippet of Data collected from android application

In [4]:
```
for i in range(3):
    data.drop(data.columns[3], axis=1, inplace=True)
```

In [5]:
```
for i in range(2):
    data.drop(data.columns[6], axis=1, inplace=True)
```

**Drop the header row**

In [6]:
```
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
```

**Drop data points that exceed 1024 data point requirement** ¶

In [7]:
```
data.drop(data.index[1024:data.shape[0]])
```

Out[7]:

| | 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

Fig. 5. Snippet of code written for data formatting

*2) Data Segmentation:* We train the CNN network proposed in Section II-A-2 for gait data extraction. For Dataset #7 and Dataset #8, the learning rate is set to 0.0001, and the number of training epochs is set to 150. Figure 6 shows four sample results obtained by the proposed network. Most of the data are correctly classified: a small portion of walking data are extracted as non-walking (red on blue) and a small portion of non-walking data are extracted as walking (red on green). The misclassification occurs at the transition area between walking and non-walking, which is reasonable since there are uncertainties for those points in the transition area. Specifically, on Dataset #7, where 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. On Dataset #8, where 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.

## B. Identification

The two datasets Dataset #1, Dataset #2, are used to evaluate the seven deep learning-based methods: CNN, CNN+LSTM, and CNN+LSTM$_{\text{fix}}$. The classification results are shown in

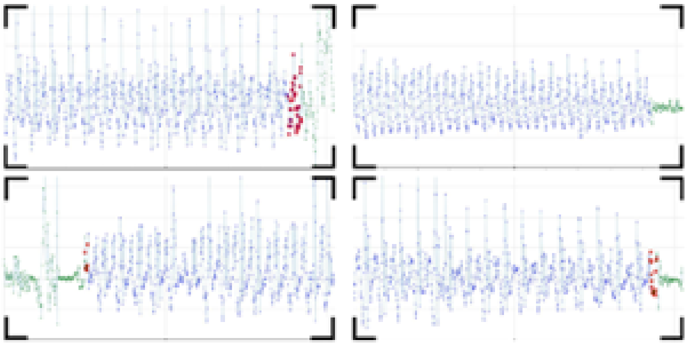

Fig. 6. Four examples of walking data extraction using the proposed method. Note that, the blue points denote the walking data, green points denote the non-walking data, and the red denotes the false classified

TABLE VI
CLASSIFICATION PERFORMANCE OF DEEP LEARNING-BASED METHODS

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

Table. All the methods achieve greater than 91.8% accuracy on Dataset #1 which contains 118 subjects, and greater than 96.7% accuracy on Dataset #2 which consists of the gait data of 20 subjects.

### C. Authentication

On Implementing the CNN and $\mathbf{CNN_{fix} + LSTM}$ architectures on datasets 5 and 6, we have received the following performance table (testing accuracies):According to these results, $\mathbf{CNN_{fix} + LSTM}$ performs more effectively on the vertical dataset (Dataset #6).

TABLE VII
AUTHENTICATION PERFORMANCE BASED ON TRAINING ACCURACY OF MODELS

| **Authentication Methods** | **Dataset #5 (Horizontal)** | **Dataset #6 (Vertical)** |
|---|---|---|
| CNN | 78.6% | 86.96% |
| CNN$_{\text{fix}}$+LSTM | 85.48% | 93.7% |

## V. DISCUSSIONS

It is thought that the three-axis accelerometer and three-axis gyroscope data, as well as their respective time stamps, are connected. As a result, the tri-axis accelerometer and gyroscope gait data was converted into six-axis inertial gait data. To process information between multiple axes, one-dimensional convolution techniques are used. For the first three layers, one-dimensional convolutional kernels of sizes 19, 13, 13 are used, with the signals being processed individually along the time zone. The suggested one-dimensional convolution kernels can ensure that the final convolution results are temporal. Because convolution is a local operation, the output features are expected to have the time series attribute as well. A 6 x 1 kernel is used to the preceding convolution results to spatially correlate the signals on multiple axes, resulting in 1 x 16 x 128 data. It is worth noting that the number of characteristics along the time axis is 16. The needed time-series is produced after rearranging it into 128 x 16. We discovered that the authentication job may be difficult while testing with the two models on datasets 5 and 6, because the 98 participants training CNN$_{fix}$ are identical for both datasets.

## VI. CONCLUSIONS

A hybrid approach that seamlessly merged the DCNN and DRNN was demonstrated for robust inertial gait feature encoding. During gait data collection, the cellphones were utilized under unconstrained settings, and no information about when, where, or how the user walks was provided. The proposed hybrid network outperformed the solo networks by a substantial margin. We were able to delve deep into machine learning frameworks like Tensorflow, as well as programming environments like Jupyter Notebook and Google Colab. Many concepts connected to the CNN and LSTM models became clearer to us. We uncovered a plethora of hyperparameters that have a direct influence on the accuracy of our model. The various lessons learned during this project have been mentioned in the "lessons learned table" which is provided in the appendix. You may find the code and details about the implementation of this project on our GitHub[1] and our website[2].

## 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

[2] D.P Kingma, J.Ba, "Adam : A method for stochastic optimization", arXiv preprint arXiv:1412.6980, 2014 - arxiv.org


[1]GitHub: https://github.com/AniIOT/ANC_FINAL_PROJECT
[2]Website: https://aniiot.github.io/ANC_FINAL_PROJECT/

This table is required and to be included in the final report
(this table does not contribute to the page count)
Please don't change the format of the table, i.e., only fill in the blanks
**end of semester reflection - lessons learned from working on the final project**
**Team # and names of team members**

**Deep Learning Based gait Recogntion using Smartphones in the wild**

| | literature (not well written or self-contained, not specific on implementation, no data source indicated, no source code indicated…) | setting up the environment and obtaining data | to have the first successful test run (issues during debugging, compatibility problems…) | obtaining results (algorithm/method is dificult to implement, hyper parameters difficult to tune...) | obtaining results (cannot duplicate what was reported in paper, if so, why?) | reporting (Intro, method, result, discussions, …) |
|---|---|---|---|---|---|---|
| specific & detailed evidence is required to support claims (e.g., links, repository sites, equation #, figure #, paragraphs, sections, etc…) | 1. Haven't added specification for preprocessing during data extraction.<br>2. Paper Link<br>3. Authors were not specific on how to implement given codes<br>4. We learnt about CNN networks as well as LSTM networks | 1. Learned to Setup Tensorflow compat V1 environment<br>2. Obtained Data from AndroSensor android app | 1. Whole project is based on Tensorflow V1 thus had to change a lot of functions<br>2. No preprocessing code provided for own dataset thus had to write our own code<br>3. No model saving code in data_extraction thus had to write our own code | 1. Data visualization code was not provided<br>2. All three networks were used by changing tensorflow environment to run tensorflow v1 instead of V2<br>3. Obtained results were accuracies of each of the network<br>4. GitHub Link<br>5. Obtainig results was the hardest part as the codes were using older vesions of the packages and also the systems we had were not good enough for the training. Thus it took us days for training | 1. Data Authentication code was incomplete so we could not obtain results for performance comparison methods.<br>2. GitHub Link | 1. The reporting tasks helped us understand a lot about writing IEEE standard reports<br>2. All the sections asked in the submission details were appropriate and very well structured |

**Team Assignment 4 - Team final project - Final Report**

**Team #18**

| **Student name: Aniruddha Anand Damle** | worked on literature | worked on implementation (data, platform, test run, debug, compatibility…) | generated results (run results, result data processing, presenting results | wrote report (Intro, method, result, discussions, …) | other significant contributions | peer approval 1 | peer approval 2 | peer approval 3 |
|---|---|---|---|---|---|---|---|---|
| specific & detailed evidence is required to support claims of contributions (make reference to specific paragrphs, equation #, figure #, code line #'s sections, etc…) | 1. Knowledge-Guided Deep Fractal Neural Networks for Human Pose Estimation<br>2. FAST-Dynamic-Vision: Detection and Tracking Dynamic Objects with Event and Depth Sensing | 1. Worked on development of android app<br>2. Developed a python script for data formatting,<br>3. Debugged and trained data extraction network | 1. Created app using MIT app inventor<br>2. Found app called Sensor_server and tried using that to collect data | 1. Wrote data extraction network method, implementation, results and discussion. | 1. Made github website<br>2. Maintained GitHub<br>3. Worked on app development | N/A | Approved | Approved |

| **Student name: Prakriti Biswas** | worked on literature | worked on implementation (data, platform, test run, debug, compatibility…) | generated results (run results, result data processing, presenting results | wrote report (Intro, method, result, discussions, …) | other significant contributions | peer approval 1 | peer approval 2 | peer approval 3 |
|---|---|---|---|---|---|---|---|---|
| specific & detailed evidence is required to support claims of contributions (make reference to specific paragrphs, equation #, figure #, code line #'s sections, etc…) | 1. Presented Numerical solution of fractal-fractional Mittag–Leffler differential<br>2. Presented paper on Dynamic Object Tracking and Masking for Visual SLAM | 1. Ran Identification code<br>2. Debugged and trained CNN Identification network<br>3. Debugged and trained CNN+LSTM network for authentication | Generated results of person identification on datasets provided by the authors | 1. Abstract<br>2. Wrote about implementing authentication section of project.<br>3. Contributed to the discussion section | 1. Collected Gait Data<br>2. Mathematical Analysis | Approved | N/A | Approved |

| **Student name: Aditya Kaduskar** | worked on literature | worked on implementation (data, platform, test run, debug, compatibility…) | generated results (run results, result data processing, presenting results | wrote report (Intro, method, result, discussions, …) | other significant contributions | peer approval 1 | peer approval 2 | peer approval 3 |
|---|---|---|---|---|---|---|---|---|
| specific & detailed evidence is required to support claims of contributions (make reference to specific paragrphs, equation #, figure #, code line #'s sections, etc…) | 1.Deep learning based Gait Recognition using Smartphones in the wild (Selected) | 1. Ran Gait Segmentation Code<br>2. Debugged and trained CNN+LSTM Identification network<br>3. Debugged and trained CNN network for authentication | Generated results of gait extraction on dataset provided by authors | 1. Wrote about implementing identification section of project.<br>2. Conclusion<br>3. Contributed to Discussion Section. | 1. Collected Gait Data<br>2. Mathematical Analysis | Approved | Approved | N/A |