RESEARCH ARTICLE OPEN ACCESS

Human Gait Analysis and Classification Using DeepLearning

Abidha R
Meena G
Pavithra E
Reshmaa B
B.E Computer Science and Engineering
A.V.C College of Engineering
Mannampandal, Mayiladuthurai

Dr Sudha.J
Associate Professor
A.V.C College of Engineering
Mannampandal, Mayiladuthurai

Abstract-**The human ankle joint interacts with the environment during ambulation to provide mobility and maintain stability. This association changes depending on the different gait patterns of day-to-day life. The main objective of this project is to predict the gait pattern whether the person will sitting or standing or standing up or sitting down or going up in the upstairs or going down in the upstairs based upon the sensor gyroscope values according to the right and left leg. Here we are using 1D convolutional neural network which provide more accuracy than any other existing models.We also compared with other Deep learning approaches such as CNN+RNN and DCNN**

Keywords-1DCNN, CNN+RNN, DCNN.

I.Introduction

The improvement and restoration of the gait performance while ambulating in daily living conditions, which involves activities of ascending/descending stairs and overground walking, has been an important aspect of the robotic rehabilitation community. Walking in a controlled lab oratory setting does not represent a realistic scenario of the real world where users involve multiple walking modes. Walking in multimodal conditions with the trivial interventions of aerobic exercises, strength training, and treadmill walking, stroke, older adults (aged 71 years and above) and Parkinson's Disease patients have shown generalizability and transference effects of gait training into unsupervised walking. Our Project focused on classification of human gait position. The process can be done by using deep learning techniques.

II. Related Work

In reference [1]This study aimed to improve the accuracy of gait event detection in the elderly using a single sensor on the waist and deep learning models. Data were gathered from elderly subjects equipped with three IMU sensors while they walked. The input taken only from the waist sensor was used to train 16 deep-learning models including a CNN, RNN, and CNN-RNN hybrid with or without the Bidirectional and Attention mechanism. The ground truth was extracted from foot IMU sensors. A fairly high accuracy of 89.73% and 90.89% was achieved by the CNN-BiGRU-Att model

reference[2] In this paper, a sequential convolution LSTM network for gait recognition using multimodal wearable inertial sensors is proposed, which is called SConvLSTM. Based on 1D-CNN and a bidirectional LSTM network, the method can automatically extract features from the raw acceleration and gyroscope signals without a manual feature design. 1D-CNN is first used to extract the high-dimensional features of the inertial sensor signals. While retaining the time-series features of the data, the dimension of the features is expanded, and the length of the feature vectors is compressed. Then, the bidirectional LSTM network is used to extract the time-series features of the data. The proposed method uses fixed-length data frames as the input and does not require gait cycle detection, which avoids the impact of cycle detection errors on the recognition accuracy.

Reference[3] gait terrain detection algorithm from the measurements of a foot-mounted Inertial Measurement Unit (IMU), using a shallow cascaded Convolutional and Long Short-Term Memory neural network (CNN-LSTM). Gait data is acquired from healthy subjects walking in an unstructured environment that includes level ground, stair ascent and stair descent. The CNN-LSTM subject-independent classifier is trained to continuously detect the terrain from the time series data, invariant to IMU initial pose.

The results show that the classifier is able to correctly detect the terrain on data from unseen subjects, in less than 90ms from toe-off (f1-score >0.89), improving further its classification performance in less than 135ms from toe-off (f1-score >0.90)

Reference[4] a gait type classification method based on deep learning using a smart insole with various sensor arrays. gait data is measured using a pressure sensor array, an acceleration sensor array, and a gyro sensor array built into a smart insole. Features of gait pattern were then extracted using a deep convolution neural network (DCNN). In order to accomplish this, measurement data of continuous gait cycle were divided into unit steps. Pre-processing of data were then performed to remove noise followed by data normalization. A feature map was then extracted by constructing an independent DCNN for data obtained from each sensor array. Each of the feature maps was then combined to form a fully connected network for gait type classification. Experimental results for seven types of gait (walking, fast walking, running, stair climbing, stair descending, hill climbing, and hill descending) showed that the proposed method provided a high classification rate of more than 90%.

In reference[5] a novel solution for gait feature extraction and gait classification. Firstly, two kinds of Two-branch Convolution Neural Network (TCNN), i.e., middle-fusion TCNN and last-fusion TCNN, to improve the correct recognition rate of gait recognition are presented. Secondly, we construct Multi-Frequency Gait Energy Images (MF-GEIs) to train the proposed TCNNs networks and then extract refined gait features using the trained TCNNs. Finally, the output of each TCNN is utilized to train an SVM gait classifier separately which will be used for gait classification and recognition. In addition, the proposed solution is measured on CASIA dataset B and OU-ISIR LP dataset.

Reference [6] The proposed approach uses the data captured by a smartphone's accelerometer and gyroscope sensors while the users perform the gait activity and optimizes the design of a recurrent neural network (RNN) to optimally learn the features that better characterize each individual. The database is composed of 15 users, and the acceleration data provided has a tri-axial format in the X-Y-Z axes. Data are pre-processed to estimate the vertical acceleration (in the direction of the gravity force). A deep recurrent neural network model consisting of LSTM cells divided into several layers and dense output layers is used for user recognition. The precision results obtained by the final architecture are above 97% in most executions. The proposed deep neural network-based architecture is tested in different scenarios to check its efficiency and robustness.

In reference[7] gait recognition algorithm is presented based on the information obtained from inertial sensors embedded in a smartphone, in particular, the accelerometers and gyroscopes typically embedded on them. The algorithm processes the signal by extracting gait cycles, which are then fed into a Recurrent Neural Network (RNN) to generate feature vectors. To optimize the accuracy of this algorithm, we apply a random grid hyperparameter selection process followed by a hand-tuning method to reach the final hyperparameter configuration. The different configurations are tested on a public database with 744 users and compared with other algorithms that were previously tested on the same database. After reaching the best-performing configuration for our algorithm, we obtain an equal error rate (EER) of 11.48% when training with only 20% of the users. Even better, when using 70% of the users for training, that value drops to 7.55%.

III. Proposed methodology

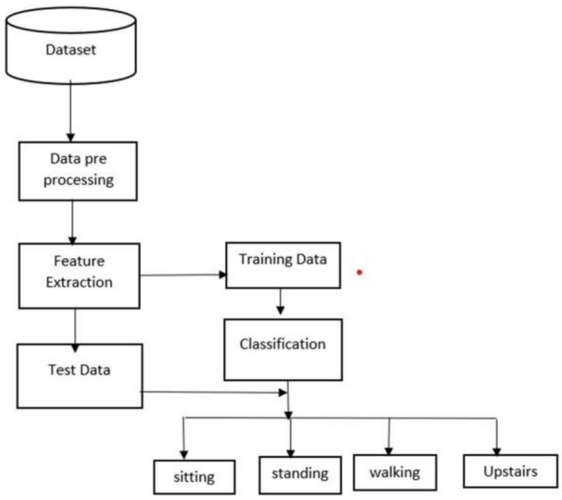
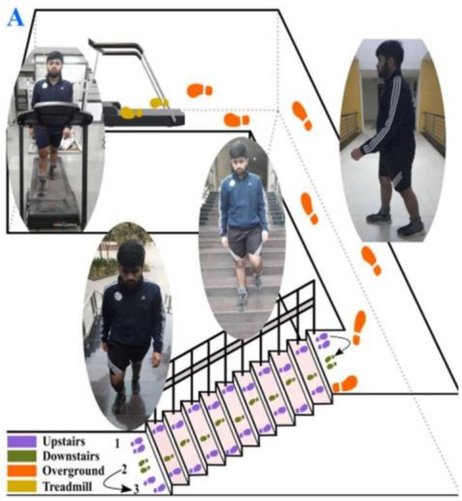


Figure 1 :Architecture of proposed model

Dataset:

The dataset will be used as training set for the training of our model. We use dataset which is collected by 40 healthy persons walking with normal speed and slowly increasing the speed to get every possible data.



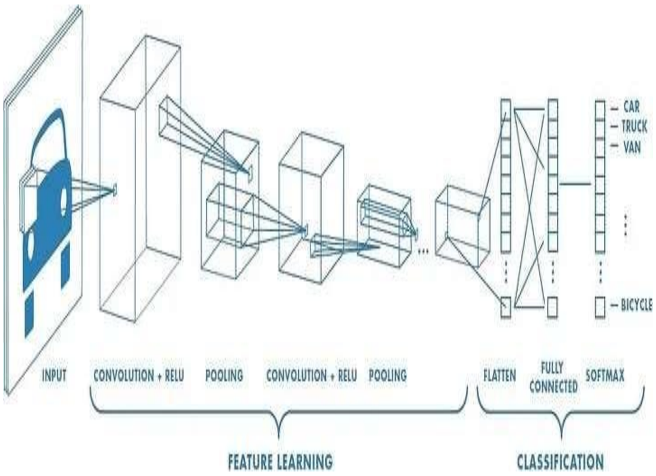


Figure 2 : CNN

There are 2 phases:

Data preparation phase:

The values obtained from the sensors should be normalized using a unit step value which is unit step segmentation. And the second step would be noise reduction ,In the 'swing phase' of the gait cycle, since one foot is separated from the ground, all eight pressure sensors for that foot should have a value of zero. But in some cases the value would be 1 even which is in the swing phase this is due to potential difference between the sensors it should be removed and changed to 0.

Development phase:

1D CNN: CNN was previously designed to operate exclusively on two-dimensional data, such as images and video, so also known as 2D CNN. Recently, in addition to CNNs that process two-dimensional data, CNNs that can process one-dimensional data have also been developed, and are commonly referred to as 1D Convolutional Neural Networks (1D CNNs) [17]. Several studies showed that for specific application, 1D
CNN is advantageous for specific applications. Several studies have shown that for certain applications 1D CNN is advantageous and thus preferable over using 2D CNN in handling 1D signals. However, it is preferable to use 1D CNN in handling 1D signals due to the following reasons [12]:

There is a significant difference in the computational complexity of 1D CNN and 2D CNN. For example, an image with dimension NxN convoluted with the KxK kernel will have a computational complexity of ~O(N^2K^2). Meanwhile, in the corresponding 1D convolution, the dimensions N and K, it is ~O(NK). This means that under equivalent conditions with the same configuration, network, and hyperparameters, the computational complexity of 1D CNN is significantly lower than 2D CNN.

POOLING : Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training the model. There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel. Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction. On the other hand, Average Pooling simply performs dimensionality reduction as a noise-suppressing mechanism. Hence, we can say that Max Pooling performs a lot better than Average Pooling.

Forward propagation of 1D CNN :

The 1D forward propagation (1D-FP) in each CNN layer is expressed as follows:

$$x^l_k = b^l_k + \sum \mathrm{conv1D}\left(w^{l-1}_{ik}, s^{l-1}_i\right)$$

Where x^l_k is input, b^l_k is the bias from neuron k^{th} in layer l, S_i^{l-1}, is output from neuron i^{th} in layer l-1, w^{l-1}_{ik} is the kernel from neuron i^{th} in layer l-1 to neuron k^{th} in layer l, conv1D (…) is used to perform "in-valid" 1D convolution without zero-padding. Therefore, the input array dimensions $x^l_k$are smaller than the output, s_i^{l-1}. The intermediate output, y_k^l can be expressed by passing the input through the activation function, x^l_k and f(.), as shown in the following equation:

ykl = f(xlk) and slk = ykl ↓ ss

Where s^l_k is the output of the k^{th} in layer l, and "↓ ss" represents the down-sampling operation with the scalar factor ss.

Backward Propagation of 1D CNN:

The BP algorithm can be brie y explained according to the following explanation. The error BP process starts from the MLP, where l and L, which denote the input and output layers equal 1. MLP is a neural network in which each neuron in the previous layer is fully connected to each neuron in the next layer [29] [30][31].Meanwhile, N_L is the number of classes in the database where the input, target, and output vectors are denoted by p, tp, and $[y_1^L,\ldots,y_N^L L]$. As a result, in the output layer for the input p, mean-squared error (MSE), E_p, can be stated as follows:

$$E_p = MSE\left(t^p, \left[y_1^L,\ldots,y_{N_L}^L\right]'\right) = \sum_{i=1}^{N_L}\left(y_i^L - t_i^p\right)^2$$

To determine the derivative of Ep by each network parameter, the delta error $\Delta^l_k = \frac{\partial E}{\partial x^l_k}$ should be calculated. Furthermore, to update the neurons bias and all the neurons weights in the previous layer, the derived chain rule is used as follows:

$$\frac{\partial E}{\partial w^{l-1}_{ik}} = \Delta^l_k y^{l-1}_i \text{ and } \frac{\partial E}{\partial b^l_k} = \Delta^l_k$$

The regular (scalar) BP of the rst MLP to the last CNN layers can be performed as follows

$$\frac{\partial E}{\partial s^l_k} = \Delta s^l_k = \sum_{i=1}^{N_{l+1}} \frac{\partial E}{\partial x^{l+1}_i}\frac{\partial x^{l+1}_i}{\partial s^l_k} = \sum_{i=1}^{N_{l+1}} \Delta^{l+1}_i w^l_{ki}$$

After the rst BP is carried out from the next layer l + 1, to the current l, the BP on the input delta of the CNN layer l is calculated as Δ^l_k. Assuming zero order up-sampled map: $us^l_k = up(s^l_k)$, hence, the delta error can be expressed as follows:

$$\Delta^l_k = \frac{\partial E}{\partial y^l_k}\frac{\partial y^l_k}{\partial x^l_k} = \frac{\partial E}{\partial us^l_k}\frac{\partial us^l_k}{\partial y^l_k} f'(x^l_k) = up(\Delta s^l_k)\beta f'(x^l_k)$$

Where $\beta = (ss)^{-1}$, hence, the delta error $\left(\Delta s^l_k \xleftarrow{\sum} \Delta^{l+1}_i\right)$ can be stated as:

$$\Delta s^l_k = \sum_{i=1}^{N_{l+1}} conv1Dz(\Delta^{l+1}_i, rev(w^l_{ki}))$$

Where rev(.) and conv1Dz(.,.) are used to back the array and perform a full 1D convolution with zero padding. The weight and bias sensitivity are stated as follows:

$$\frac{\partial E}{\partial w_{ik}} = conv1D\left(s^l_k, \Delta^{l+1}_i\right) \text{ and } \frac{\partial E}{\partial b_{lk}} = \sum_n \Delta_k(n)$$

IV.Results and Discussion

Evaluation technique:

Hence the dataset is not balanced we use the performance metrics Precision, Recall, F1 score. And the result we obsereved is,

MODEL	PRECISION	RECALL	F1
CNN	0.6227	0.6583	0.9200

MODEL	ACCURACY
1D CNN	0.9466

References

[1] Trommel, R. P., Harmanny, R. I. A., Cifola, L., & Driessen, J. N. (2016, October). Multi-target human gait classification using deep convolutional neural networks on micro-Doppler spectrograms.

[2]Jun, K., Lee, S., Lee, D. W., & Kim, M. S. (2021). Deep learning-based multimodal abnormal gait classification using a 3D skeleton and plantar foot pressure. *IEEE Access*, *9*, 161576-161589.

[3] Mogan, J. N., Lee, C. P., & Lim, K. M. (2022). Advances in Vision-Based Gait Recognition: From Handcrafted to Deep Learning. *Sensors*, *22*(15), 5682.

[4] Filipi Gonçalves dos Santos, C., Oliveira, D. D. S., A. Passos, L., Gonçalves Pires, R., Felipe Silva Santos, D.,

Pascotti Valem, L., ... & Colombo, D. (2022). Gait recognition based on deep learning: A survey. *ACM Computing Surveys (CSUR)*, *55*(2), 1-34.
[5] Kaur, R., Motl, R. W., Sowers, R., & Hernandez, M. E. (2022). A Vision-Based Framework for Predicting Multiple Sclerosis and Parkinson's Disease Gait Dysfunctions-A Deep Learning Approach.
[6] Rana, S. P., Dey, M., Ghavami, M., & Dudley, S. (2022). Markerless gait classification employing 3D IR-UWB physiological motion sensing.
[7] Khan, M. A., Arshad, H., Damaševičius, R., Alqahtani, A., Alsubai, S., Binbusayyis, A., ... & Kang, B. G. (2022). Human gait analysis: A sequential framework of lightweight deep learning
[8] Khan, A., Khan, M. A., Javed, M. Y., Alhaisoni, M., Tariq, U., Kadry, S., ... & Nam, Y. (2022). Human gait recognition using deep learning and improved ant colony optimization.
[9] Shi, L. F., Liu, Z. Y., Zhou, K. J., Shi, Y., & Jing, X. (2023). Novel Deep Learning Network for Gait Recognition Using Multimodal Inertial Sensors.
[10] Thuan, N. D., & Hong, H. S. (2022, November). Wireless Smart Shoes for Running Gait Analysis Based on Deep Learning.
[11] Choi, J. O., & Rha, D. W. Automated gait classification in children with bilateral spastic cerebral palsy using Markerless Motion Capture and Deep Learning algorithm.