Received: 20 June 2022 | Revised: 25 September 2022 | Accepted: 26 September 2022

IET Biometrics

DOI: 10.1049/bme2.12103

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH

# A robust covariate-invariant gait recognition based on pose features

**Anubha Parashar**[1] | **Apoorva Parashar**[2] | **Rajveer Singh Shekhawat**[1]

[1]Computer Science and Engineering, Manipal University Jaipur, Jaipur, India

[2]Computer Science and Engineering, Maharshi Dayanand University, Jaipur, India

**Correspondence**

Anubha Parashar, Computer Science and Engineering, Manipal University Jaipur, Jaipur, India.
Email: anubhaparashar1025@gmail.com

**Abstract**

Gait recognition uses video of human gait processed by computer vision methods to identify people based on walking style. The complexity introduced by covariates makes the previous methods less efficient and inaccurate. This study proposes an approach based on pose features to attempt gait recognition of people with an overcoat, carrying objects, or other covariates. It aims to estimate human locomotion using Convolutional Neural Networks. Gathering video data, extracting video frames in a particular order, posture estimation for each frame, using multilayer RNN for gait recognition from the pose, and obtaining one-dimensional object vectors, are all critical steps. Furthermore, these one-dimensional identification vectors are stored in a data set along with the name of the person walking in the video. The proposed data set is used to train a classification model to predict the person in a new video by first processing it to get its identification vector and then to use it as a test case in the classification model. A graphical user interface was also developed so that anyone with no programming or technical experience can easily use the tool. The developed application does everything for gait detection from mp4 videos by obtaining the identification vectors and saving them into the data set. Using this application, one can quickly identify the person walking in a video. The results obtained offered an accuracy from 60.88% to 95.23%.

**KEYWORDS**

biometrics, covariates, deep learning, gait recognition, pose estimation

## 1 | INTRODUCTION

Surveillance systems (CCTV) combined with biometric systems is the most popular approach in vogue to ensure security and privacy of people. Surveillance Systems are installed almost everywhere because they help in overseeing cash registers and employees in secure premises and the people in protected environments, by screening their movement in high-risk area, keeping an eye on unwanted visitors strolling around. Biometrics have their some inherent advantages as compared to manual watch. It can be utilised for identification of criminals, present a superior level of security than usual means of authentication [38]. These systems can be integrated in surveillance systems by embedding biometric feature recognition of gait. Gait can be defined as a person's manner of walking [1]. Human gait refers to locomotion achieved through the movement of human limbs [2]. Human gait analysis, refers to the systematic study of human locomotion and get beneficiary data or study out of it. In this research, gait recognition has been implemented so that a set of values can be obtained from a gait study over the video of a person walking in the frame. Thus, we create an application that can be used with existing surveillance systems, which uses gait recognition to make it smarter and more reliable. In comparison with other biometric features like face, voice or fingerprint, it has been found that gait can be reliably perceived at a greater distance with simple instrumentation, and it does not require the cooperation or awareness of the individual.

There a large amount of data that is being generated every day and stored. There has been a huge growth of both

structured and unstructured data. The amount of data produced every day in 2022 is massive in comparison to the data produced a decade ago, and this has resulted in an increase and use of machine learning and neural networks. But, most of this data produced is Unstructured Data. That means, most of the data produced today is in the form of e-mails, documents, videos, photos, audio files, presentations, webpages and many other kinds of unstructured data. Unstructured data can be seen or defined as the data that is not actively managed or stored in a structured format like, RDBMS, excel etc. In this case, we will be working with video data. The number of CCTV cameras used by police in India has significantly increased in last 4 years [3].

There are few disadvantages of unstructured video data is:

(1) It is not easy to store, manage, or preserve unstructured data, because of the absence of structure and schema.
(2) It is very difficult to index or tag the data.
(3) Operations-like update, delete, and search is very difficult, due to unclear structure of a large video data set.
(4) Compared to structured data, storage cost is high. For example, most of the CCTV data are stored for a particular amount of time (e.g. A month) and then deleted due to lack of storage.

Processing this unstructured kind of data to get valuable information out of it is not an easy task. We implement a deep neural network to identify characteristics of a person's manner of walking known as a person's gait. And to do that, we implement pose estimation before actual gait estimation. Human pose estimation is a computer vision problem, that deals with the prediction of the body part and joint positions of a human [4]. Through pose estimation on a video, we are able to track every movement of a human and do various biomechanical analysis and study over a period of time. Gait recognition techniques are being studied and various theoretical methodologies have been proposed. The only constraints that bind the practical application to this study are covariates, processing capacity, and robustness of the algorithm [5, 6]. The accuracy of gait recognition is hampered by the presence of covariates, which play a very crucial role in the process. The viewpoint, the clothing, the footwear, the kind of surface, the carried weight, the walk pace, the amount of time, and the emotional state are all covariates [1].

### 1.1 | Motivation

Video surveillance has become a vital aspect of security systems in numerous enterprises. Inside commercial buildings, college campuses, hostels, and residential ventures, CCTV cameras can be found practically everywhere, and all of these surveillance systems have control rooms that keep the records/CCTV footage. However, these footages are unstructured data that must be inspected manually in the event of an emergency with no technological assistance. The goal is to develop a tool that can be used to employ gait analysis to make this procedure smarter [7]. We can save gait analysis data and use it to forecast the same individual in other videos later. A basic graphical user interface (GUI) should be supplied to make the application easy to use and accomplish numerous important tasks with the click of a button. The current system has a number of drawbacks:

(1) The data obtained from surveillance Systems are unstructured and raw, and cannot be used in any kind of neural network or ML model [8].
(2) The surveillance footage must be manually to identify people which can turn into a lengthy process.
(3) Surveillance systems have been the same since a very long time, with no software advances.
(4) Covariate situation is not handled properly [9–11].

### 1.2 | Contribution

The main contributions of the paper are:

1. Pose-based GEIs can better represent body parts and dynamics descriptors with respect to the usually blurred depiction provided by a general GEI (Gait Energy Image).
2. Pose based features for better handling the covariates. This makes the processing very robust against various covariate factors such as clothing, carrying conditions, shoe types and so on.
3. The necessary gait data (set of values) can be stored and preserved, even when the video data is deleted.
4. Creation of new gait data set for further train and test for making robust gait recognition system.

### 1.3 | Organisation

In section 1, we summarises the theoretical concepts used and implemented in the paper. Section 2 discuss the work that is related. In Section 3, we presents the methodology that was done and the concepts and architecture that was chosen. Section 4 tells the implementation that was adopted in detail with enough pictorial assistance to understand the whole architecture better. It explains what modules were created to implement the whole architecture and dives into the various functions and classes created. It also discusses the result of the software mainly the accuracy of the classification that will tell us how accurately the gait recognitions are done. Section 6 conveys the conclusion and the possible future scope of the project.

## 2 | LITERATURE REVIEW

Variation in viewing angles, clothing, and carrying conditions degrade the gait recognition rate. Model-based approaches are quite efficient in handling such variations. Weizhi et al. [4] proposed a three-dimensional convolution neural network and LSTM network to capture spatial and temporal features from two-dimensional images. Their model is good at capturing

spatial and temporal features. The three-dimensional pose-based method can extract features more precisely in view variation than two-dimensional pose-based methods. But, their model has very low accuracy. Initial techniques to recognise gait were mostly based on appearance. In appearance-based techniques, parameters were taken from silhouettes. It is easy to compute the silhouettes and gives good recognition accuracy but gets highly affected due to covariates. In contrast to appearance-based, model-based approaches are not much affected due to covariates but require a high computational cost and do not give good recognition accuracy with low-resolution images. Liao et al. [12] proposed a model based on pose features that use CNN for gait recognition of 3-dimensional human pose. Since their model uses 3-dimensional pose estimates, it is invariant to changes in the view angles and various factors. Also, using a three-dimensional pose for spatial, temporal features improves the accuracy of the model. Their proposed model provides an effective representation of gait features and robustness towards covariate condition variations. But their model has very low accuracy and a high computational cost as it converts two-dimensional joint points to three-dimensional ones. There is a big fluctuation in the accuracy when the angle difference between train and test set becomes 90°.

Early techniques do not consider the entire height of the subject, due to which there is a decrease in recognition accuracy. Sokolova et al. [13] considered the entire height, along with the joint angle points. They proposed a novel pose-based CNN approach that considers the entire height of a silhouette and joint angles. Their method is not much affected due to covariates, especially viewing angles. Their consideration of the entire height and the joint angle points improves the recognition accuracy instead of considering the entire silhouettes. Their model has a lower recognition rate because their technique only uses motion whilst eliminating all the colour information. Tavares et al. [14] proposed the PifPaf technique that extracts features from noisy environments with noises. They are using images from high-resolution cameras and thus, are getting the best pose features. Their model has given poor results and has low accuracy. Also, they did not consider covariate situations. Their proposed model is costly as it requires high-resolution cameras, and high-resolution cameras are not installed at all places.

Luo et al. [15] targeted difficulties in real three-dimensional structured data and proposed a hierarchical temporal memory network using convolution neural network and recurrent neural network. Their proposed method uses an estimation technique to detect body shape, images of body-parsing, and virtual garments. Due to the use of all these techniques, their model performs well in object and clothing variations. Their method captures both spatial and temporal features which is more time-consuming. This algorithm performs well only on large-sized data sets. Hossen et al. [16] proposed a RNN-based method. Recurrent neural networks with gated recurrent units architecture are very powerful in capturing the temporal dynamics pose sequence of the human body and performing recognition. The authors also designed a low-dimensional gait feature descriptor based on the 2D coordinates of human pose information proven to be invariant to various covariate factors and effective in representing the dynamics of various gait patterns. Their results are robust as they found effective gait features for the view variant environment. Nevertheless, they did not consider the cross-view conditions.

Current methods based on skeleton have achieved less accuracy as they tackle normal and noise data while recognising gait. In ref. [17], the authors proposed a method based on skeleton using the Siamese network along with autoencoder networks. Their model is effective for covariates like variations in clothes, time, and carrying objects. Their method can also reconstruct common trajectories and perform accurate gait recognition with images with side view angles. They are not dealing with the cross-view condition. Li et al. [18] proposed a model based on a joint relationship mapping pyramid to capture spatial and temporal features. The computational cost of their approach is very high. Kooksung et al. [19] proposed a new approach for automatically extracting the features with the help of an autoencoder based on RNN. Their proposed method is better at recurrent neural network features than principal component analysis and singular value decomposition. Their recurrent neural network performance is very low because they did not consider sequential data while decreasing dimensionality.

## 3 | METHODOLOGY

The application uses three major concepts of Object detection [20], Pose Estimation [21], and Gait recognition [22].

### 3.1 | Object detection

Object detection is a computer vision and image processing task that involves recognising, identifying, and finding occurrences of meaningful things (in our case, people) in images and videos. R-CNN [23] and variations like Fast R-CNN [24] and Faster R-CNN [25] are the most common object detection algorithms, followed by Single Shot Detectors (SSDs) [26] and YOLO [27]. R-CNNs are one of the few object detectors based on deep learning and neural networks. While R-CNN and its derivatives are very accurate, they are also quite sluggish. Both Single Shot Detectors (SSDs) and YOLO employ a one-stage detector technique to help boost the speed of deep learning-based object detectors. This implies that only one trip through the neural network is required to forecast all of the bounding boxes (objects) in one go. Rather than employing CNN, these algorithms tackle the issue as a regression problem. We all are working with YOLOv3 (version 3) [28], specifically YOLO trained on the COCO data set [29].

### 3.2 | Pose estimation

The challenge of distinguishing human positions is a challenging one for computer vision. The algorithm must cope

with a vast number of conceivable human positions, as well as different changes in human appearance (such as clothes) and the presence of a large number of individuals in close vicinity. CNN (Convolutional Neural Network) [30] have recently been proved to be extraordinarily resilient in their ability to do these tasks end-to-end. Human posture estimate is performed by the architecture, as detailed here. The following body joints and components are important in pose estimation and must be detected:

1. Right ankle
2. Right knee
3. Right hip
4. Left hip
5. Left knee
6. Left ankle
7. Pelvis
8. Thorax
9. Upper neck
10. Head top
11. Right wrist
12. Right elbow
13. Right shoulder
14. Left shoulder
15. Left elbow
16. Left wrist

**FIGURE 1** Pose estimation.

Figure 1 depicts how gait information is saved and can be seen how covariates like bags and coats effects the pose features. In Figure 6, the algorithm proposes to extract pose features first and then use it for gait analysis, and focuses just on the pose of the person and changes in the pose of the person.

As shown in Figure 2, the CNN flow comprises two linked deep neural networks. The network in the illustration is a detection network that uses a per-pixel sigmoid loss to recognise distinct body parts. Because we want to detect the 16 body joints and components specified above, the result is a collection of 16 feature points. This technique of detecting body parts is quite reliable, and the locations of these body parts are subsequently plotted using lines.

A CNN with two components is the suggested architecture. The first component is a deep network for human body detection. Using a pixel-wise sigmoid cross-entropy loss function, we train detectors together. The second component is a deep LSTM network that determines where all pieces are located (both visible and occluded). The suggested model directs the network's attention and encodes structural part connections.

## 3.3 | Gait recognition

We plan to implement gait recognition using pose estimation. The spatial features can be defined as features that provide locations of various natural or artificial boundaries or shapes to

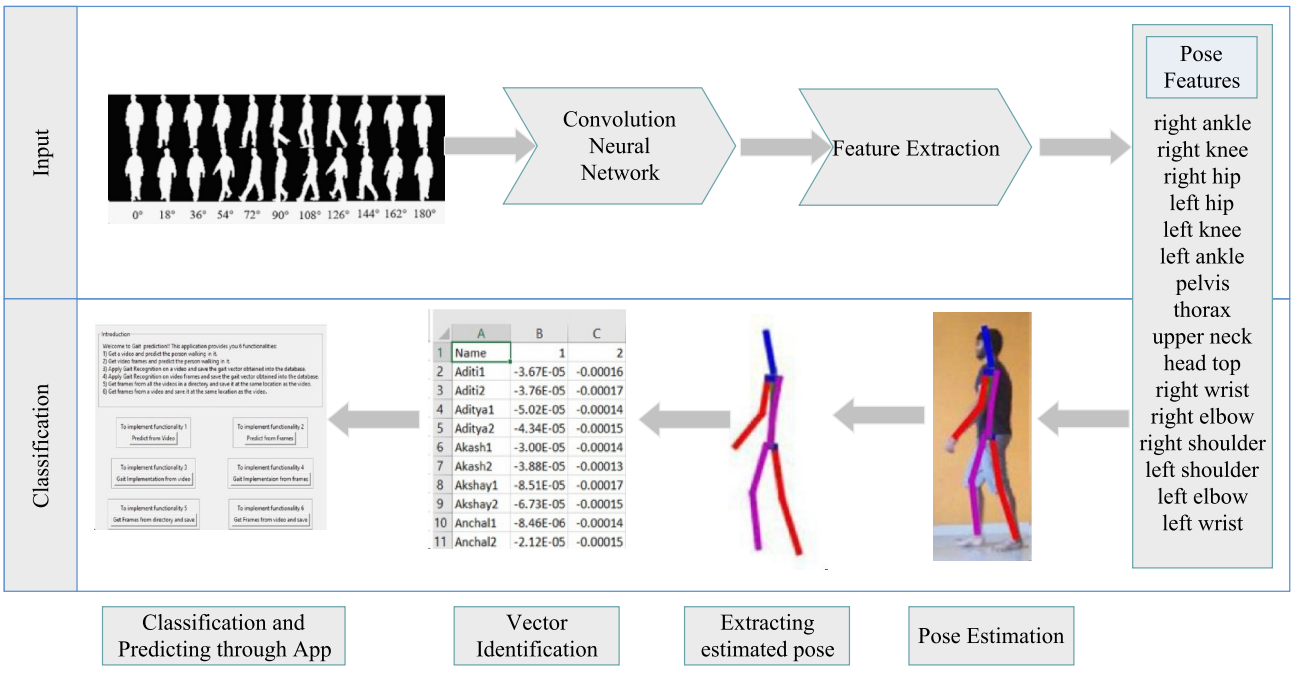



**FIGURE 2** Human pose estimation via convolution neural network.

help visualise spatial data. Spatial refers to space. The spatial features obtained from pose estimation are to be used to get gait features of a person. As we have divided the video into multiple frames and then implemented pose estimation on each frame (image), we have to compare adjacent (previous in our case) spatial descriptors to extract temporal features from a set of pose descriptors for each frame. Temporal refers to time. And as we extract temporal features from a set of spatial features, the spatiotemporal are obtained, which is basically analysis data collected across both time and space. This is done using LSTM [31].

## 4 | IMPLEMENTATION AND RESULTS

This section will provide the detailed step-by-step methodology that was used to create the application.

### 4.1 | Obtaining video data

Data is the most important aspect of any study related to analysis. As we are working with the way a person walks, it is necessary to get enough sample videos of different people walking so that all the processes can be executed with actual practical application, so that we could a see the results of every step or phase. Videos of a few people walking across the frame must be obtained to work upon. The person should be fully visible in the video.

### 4.2 | Extracting frames in required format

After getting the data, it is necessary to get it into a format that can be used in further algorithms. And the first step is, to get multiple frames out of the video. Though, it is possible to just divide the video into standard frames (figure 3) like screenshotting the video after each small-time interval, but the video can have other objects in the video, which will pose as a difficulty and would be difficult to ignore or discard in further processes. So, it is very necessary to crop the image so that the frames contain only the person walking. And to do that, it is most preferable to place the person at the centre of the cropped frame.

Initially, frames are extracted using cv2 functions and for each iteration for getting a frame, additional code is added to use YOLOv3 object detection algorithm (figure 4), which detects the person in the video. The frame is then be cropped and square-fitted accordingly, so that we finally obtain a square framed image which has the person at its centre at that instance. The code should convert the frames into required pixel dimensions before saving it to a location which can be accessed later on. We used YOLOv3 (version 3), in particular YOLO trained on COCO data set.

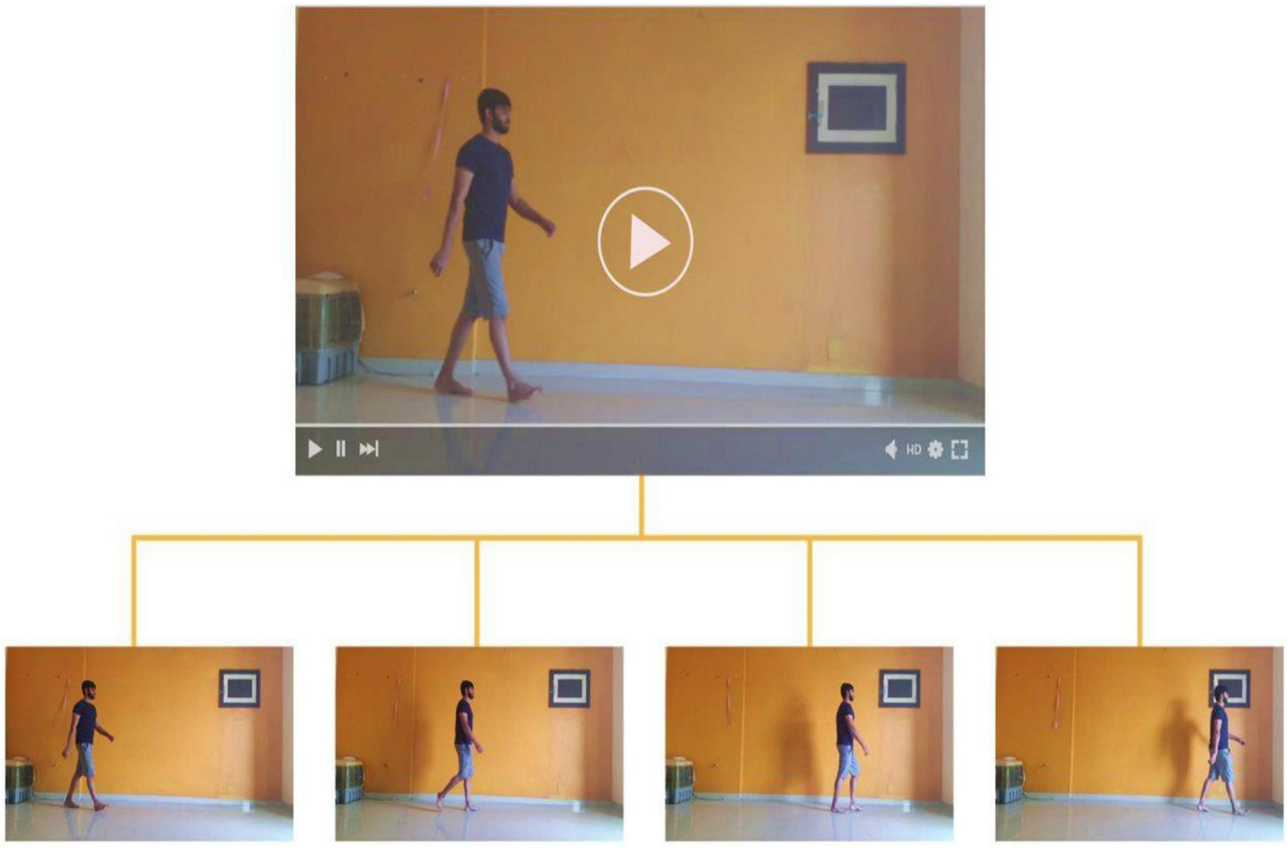


**FIGURE 3** Dividing video into frames.

## 4.3 | Pose estimation

After getting the appropriate frames out of the video, the video frames can then be passed on to the module that performs the pose estimation (figure 5). To do this, frames of a video is read from the file system and then stacked into a 4d array. This array is then passed to the pose estimation module that will perform pose estimation maintaining the order of the frames. Algorithm for pose estimation is coded so that the pose in each frame can be obtained. The CNN trained to detect the individual body parts using a per-pixel sigmoid loss. Its output is a set of feature points.

The generated detection for both visible (neck, head, left knee) and occluded (ankle, wrist, right knee) components is shown in the first row of Figure 6. (drawn with a line). It is worth noting that the occluded sections' confidence is substantially lower than the non-occluded parts' but still greater than the backdrop, giving important information regarding their approximate placement. The result of the background removed images is shown in the second row. Notice how the

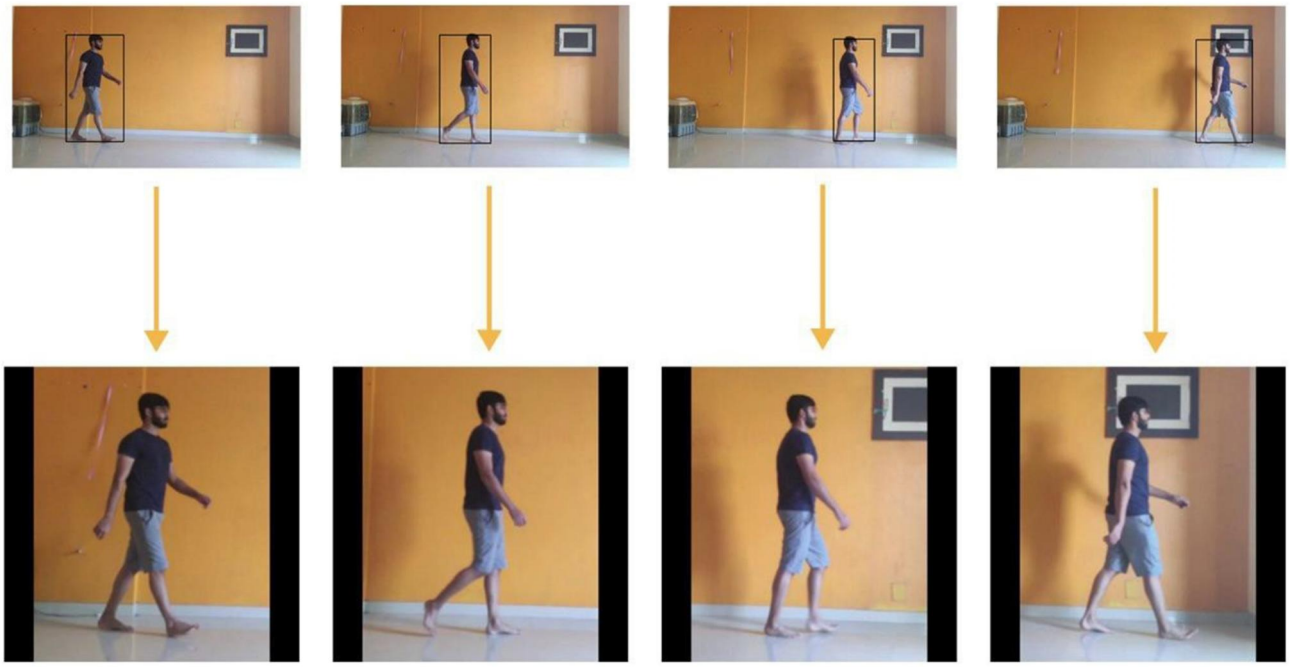


**FIGURE 4** Using human detection to crop and square-fit the frame as required.

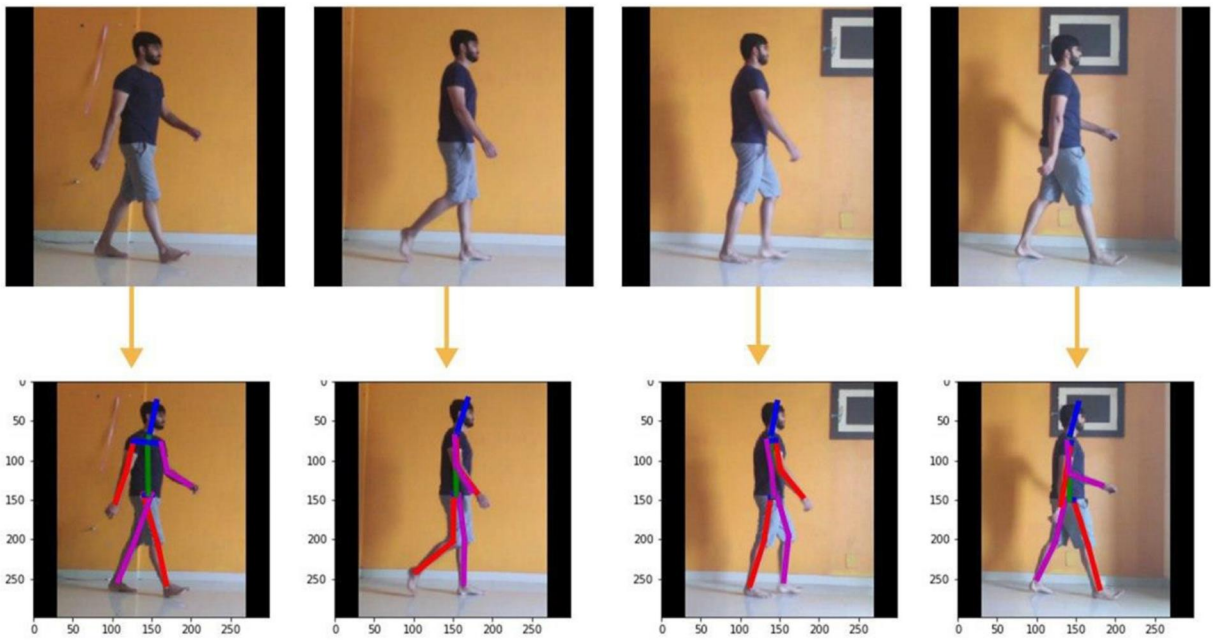



**FIGURE 5** Pose estimation on each frame.

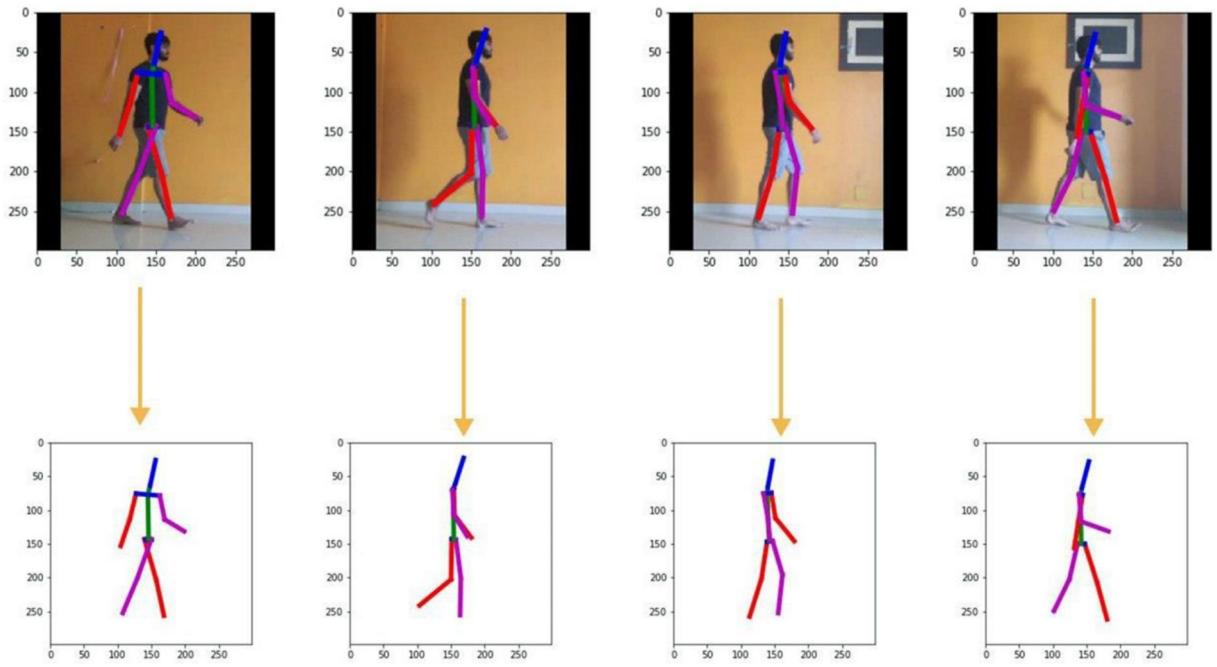


**FIGURE 6** Extracting the estimated pose.

confidence for viewable portions is greater and more localised, indicating that the network is capable of providing high confidence for the right position of occluded components. The descriptors that incorporate stance characteristics are used to extract spatial data from video frames.

## 4.4 | Gait recognition

From the pose estimation, we get a structural view of how the person walks and gives us numerical structural positions and spatial features of the person after each small interval in the walking. The second sub-network module that implements gait is in charge of using a convolutional network to further convert the obtained spatial information into one-dimensional posture descriptors. Each layer in classic neural networks feeds into the next. Each layer feeds into the next tier and straight into the levels 2–3 hops away in a network with leftover blocks. We now transmit these posture descriptors to several multilayer recurrent cells after preparing our data for the primary procedure that will execute gait analysis. This is required to compare the several instances we collected as frames and get the temporal characteristics, which are essentially analysis data on the change in attitude of the individual as he moves over time. We use the LSTM to get the temporal (spaciotemporal) features (Figure 7). All temporal features are finally aggregated with Average temporal pooling into one-dimensional identification vector with good discriminatory properties. This one-dimensional identification vector contains all the necessary differential features as values that can be saved into a data set and/or used for identifying or predicting the person. We use a pretrained model 'H3.6m-GRU-1.ckpt', trained on the data set 'AVA Multi-View Data set for Gait Recognition (AVAMVG)'.

## 4.5 | Creating data set

After having a complete implementation of gait analysis, where the gait identification vector (Figure 8) can be obtained from the video of a person walking, this identification vector is just the product of processing the video data (which is an unstructured data form). This identification vector values (structured data form) can be saved into a database or used for prediction. So, after extracting gait features and obtaining the identification vector, it should be stored in a database (excel or CSV) with the name of the person who is walking in the video. And as we save multiple gait identification vector values into the data set, we will have a database that can be used to train a classification model.

## 4.6 | Predicting person in a new video using SVM classifier

Using the database created in the previous step, we can train a classification model. Classification is a process of categorising a given set of data into classes, and in our case, every person in the database is a class. The classification program (or function) needs to be coded in such way that it takes a video or a set of frames as input and passes it to steps and get the identification vector for the input video. This identification vector can then be used as a test data to get the classification (prediction) of who is walking in the new video. A Support Vector Machine (SVM) classification model will be idle for this classification problem as it is observed that an identification vector contains 1536 distinct values that describe the gait of a person. The data from the database csv and the new identification vector needs to be feature scaled which is a requirement of the SVM classifier.

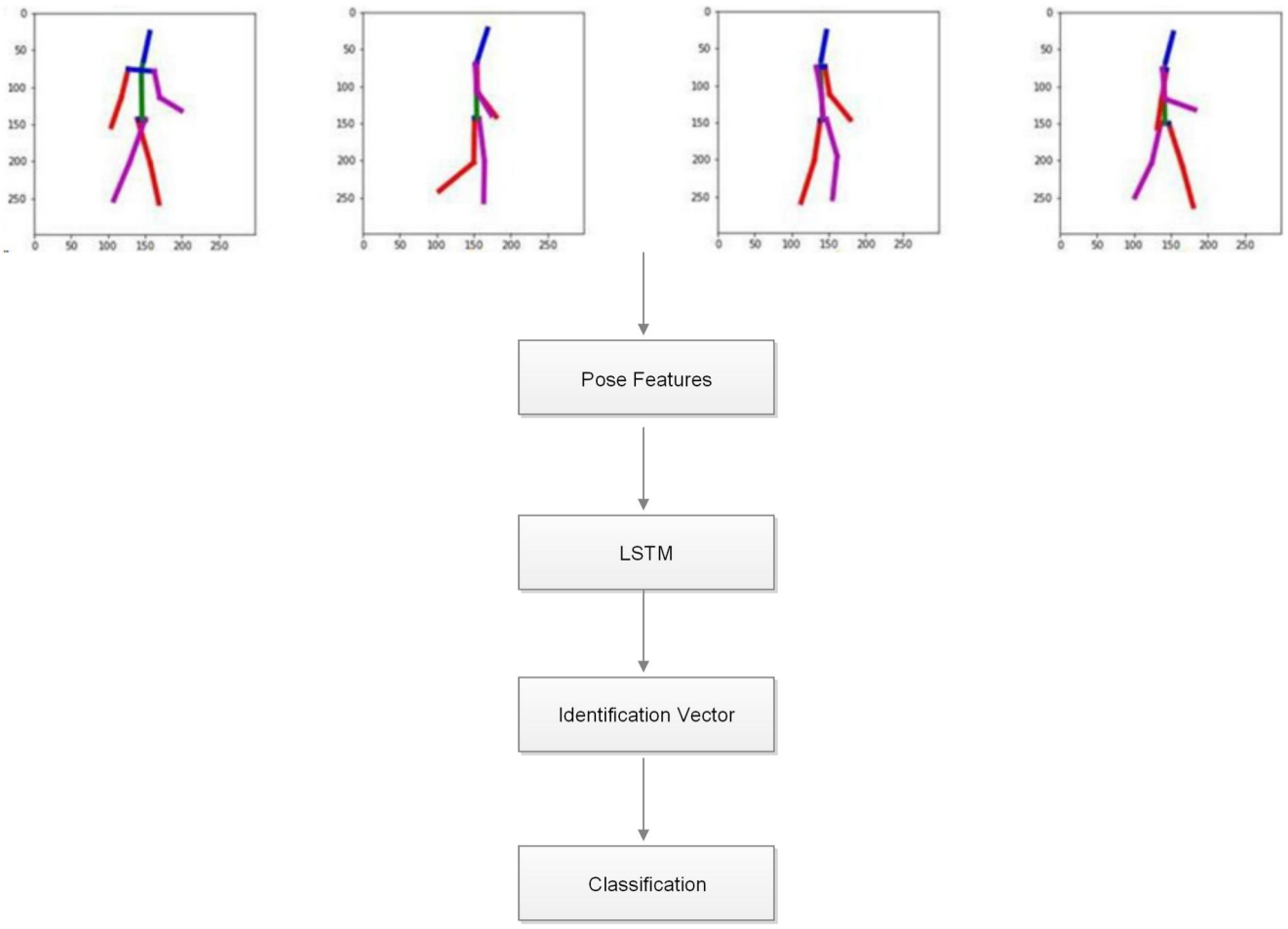



**FIGURE 7** Gait recognition after Pose Estimation.

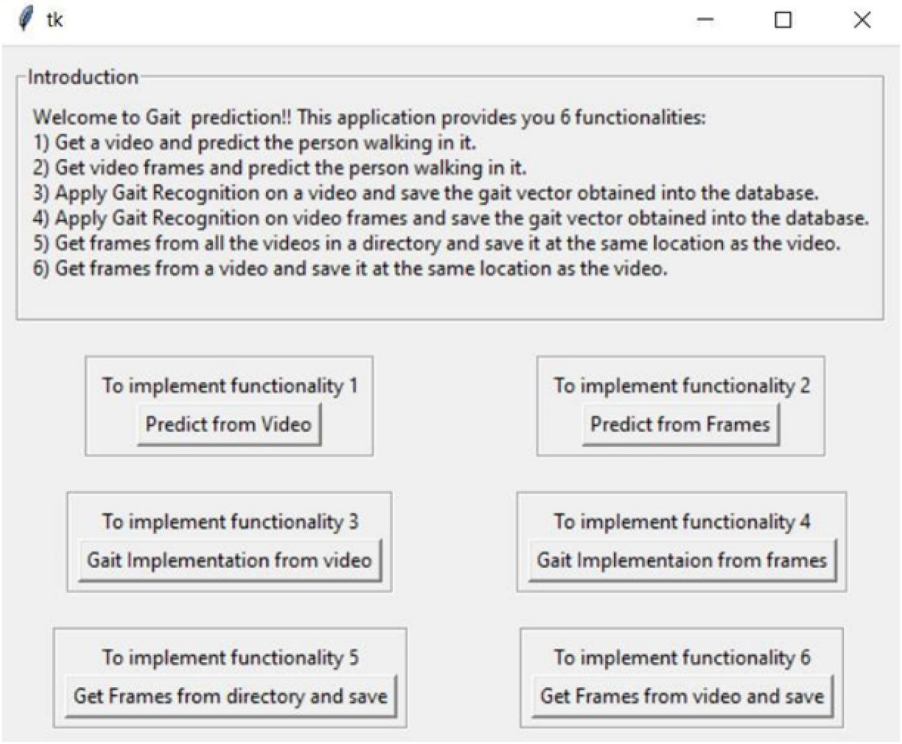
| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Name | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 2 | Aditi1 | -3.67E-05 | -0.00016 | -5.58E-05 | -0.0001 | -0.00029 | 0.000418 | 4.45E-05 | 4.62E-05 | 0.000317 | 0.000262 | 0.000239 | 3.02E-05 |
| 3 | Aditi2 | -3.76E-05 | -0.00017 | -4.15E-05 | -0.0001 | -0.00028 | 0.000418 | 4.89E-05 | 4.26E-05 | 0.000308 | 0.000267 | 0.000243 | 2.85E-05 |
| 4 | Aditya1 | -5.02E-05 | -0.00014 | -7.23E-05 | -8.05E-05 | -0.0003 | 0.000431 | 3.56E-05 | 5.20E-05 | 0.000337 | 0.00024 | 0.000222 | 2.00E-05 |
| 5 | Aditya2 | -4.34E-05 | -0.00015 | -6.28E-05 | -8.27E-05 | -0.0003 | 0.000447 | 4.26E-05 | 5.08E-05 | 0.00032 | 0.000244 | 0.000228 | 2.56E-05 |
| 6 | Akash1 | -3.00E-05 | -0.00014 | -6.05E-05 | -9.09E-05 | -0.00028 | 0.000441 | 3.62E-05 | 4.98E-05 | 0.000319 | 0.000252 | 0.000231 | 4.35E-05 |
| 7 | Akash2 | -3.88E-05 | -0.00013 | -5.67E-05 | -9.00E-05 | -0.00028 | 0.00044 | 3.02E-05 | 4.35E-05 | 0.000317 | 0.000261 | 0.000227 | 3.25E-05 |
| 8 | Akshay1 | -8.51E-05 | -0.00017 | -5.36E-05 | -7.54E-05 | -0.0003 | 0.000435 | 6.09E-05 | 5.99E-05 | 0.000328 | 0.00033 | 0.000264 | -1.39E-05 |
| 9 | Akshay2 | -6.73E-05 | -0.00015 | -7.15E-05 | -7.90E-05 | -0.00026 | 0.000415 | 6.45E-05 | 5.87E-05 | 0.000323 | 0.000292 | 0.000244 | 2.05E-05 |
| 10 | Anchal1 | -8.46E-06 | -0.00014 | -4.20E-05 | -9.58E-05 | -0.00025 | 0.000432 | 3.03E-05 | 4.73E-05 | 0.000283 | 0.000266 | 0.000237 | 2.77E-05 |
| 11 | Anchal2 | -2.12E-05 | -0.00015 | -2.76E-05 | -1.00E-04 | -0.00026 | 0.000416 | 3.88E-05 | 3.65E-05 | 0.00029 | 0.000286 | 0.00025 | 2.15E-05 |
| 12 | Ayush1 | -3.55E-05 | -0.00015 | -3.97E-05 | -7.02E-05 | -0.00029 | 0.000454 | 3.85E-05 | 3.02E-05 | 0.000306 | 0.000264 | 0.000242 | 2.09E-05 |
| 13 | Ayush2 | -2.85E-05 | -0.00016 | -3.96E-05 | -7.63E-05 | -0.00027 | 0.000446 | 4.95E-05 | 4.73E-05 | 0.000283 | 0.000273 | 0.000245 | 4.03E-05 |
| 14 | Nutan1 | -3.67E-05 | -0.00014 | -4.53E-05 | -9.79E-05 | -0.00024 | 0.000411 | 3.35E-05 | 4.51E-05 | 0.000291 | 0.000258 | 0.000236 | 3.80E-05 |
| 15 | Nutan2 | -4.17E-05 | -0.00015 | -5.08E-05 | -9.50E-05 | -0.00025 | 0.000421 | 4.96E-05 | 5.10E-05 | 0.000288 | 0.000276 | 0.00023 | 4.94E-05 |
| 16 | Santosh1 | -2.05E-05 | -0.00016 | -5.94E-05 | -7.34E-05 | -0.00031 | 0.000445 | 3.43E-05 | 5.29E-05 | 0.00032 | 0.000251 | 0.000227 | 1.57E-05 |
| 17 | Santosh2 | -2.57E-05 | -0.00016 | -6.44E-05 | -8.09E-05 | -0.00029 | 0.00046 | 4.87E-05 | 6.77E-05 | 0.000301 | 0.00025 | 0.00023 | 3.33E-05 |
| 18 | Sucheta1 | -4.13E-05 | -0.00017 | -2.13E-05 | -8.18E-05 | -0.00028 | 0.000418 | 4.78E-05 | 5.96E-05 | 0.000319 | 0.000268 | 0.000246 | -2.12E-05 |
| 19 | Sucheta2 | -5.21E-05 | -0.00017 | -2.72E-05 | -8.28E-05 | -0.00029 | 0.000413 | 6.57E-05 | 6.76E-05 | 0.000318 | 0.000265 | 0.000247 | -1.63E-05 |
| 20 | Vivek1 | -4.39E-05 | -0.00014 | -6.73E-05 | -9.14E-05 | -0.00027 | 0.000404 | 5.41E-05 | 5.71E-05 | 0.000308 | 0.000234 | 0.000217 | 1.14E-05 |
| 21 | Vivek2 | -6.33E-05 | -0.00015 | -4.87E-05 | -8.82E-05 | -0.00029 | 0.000426 | 5.14E-05 | 4.74E-05 | 0.000314 | 0.000267 | 0.000237 | 1.25E-05 |

**FIGURE 8** Sample of how the application creates a database csv.

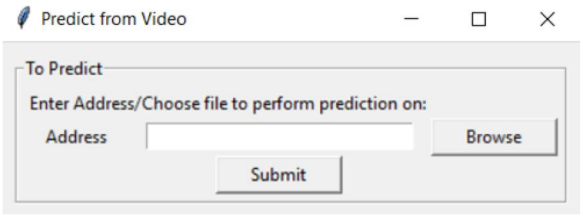
## 4.7 | Designing a user interface

The program is meant to make surveillance systems smarter, and a person who is handling the surveillance, need not have any programming or technical knowledge. So, a simple GUI (Figure 9) is designed so all these processes can be put together and be easily used by anyone.

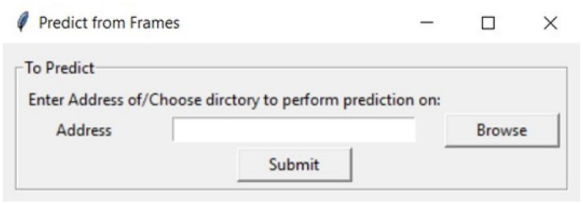
We made sure that every necessary function can be implemented at the click of a button. The GUI describes about the functionalities, so that it helps a person operate the



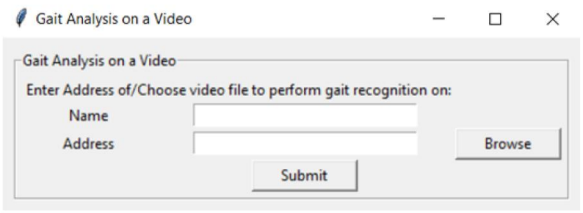

**(a)** Graphical User Interface for the program.



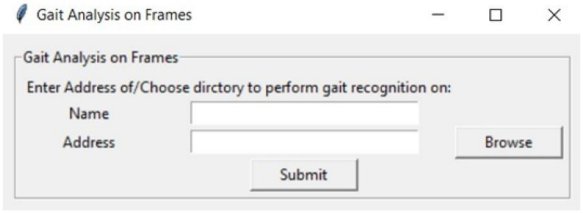

**(b)** Predict from Video in GUI.



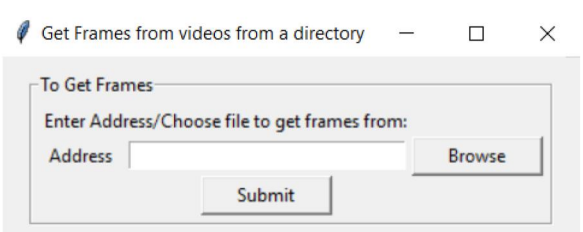

**(c)** Predict from Frames in GUI.



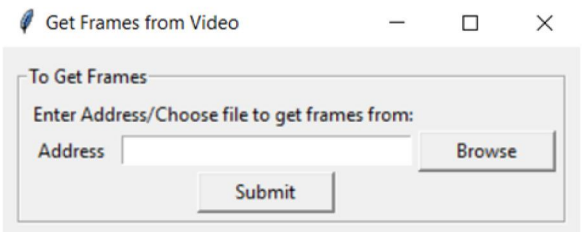

**(d)** Gait Analysis on a Video in GUI




**(e)** Gait Analysis on Frames in GUI




**(f)** Get frames from a directory




**(g)** Get Frames from Video in GUI

**FIGURE 9** Application interface for surveillance system.

application easily. Some important functionalities of this application that are available in the GUI are:

(1) Get a video and predict the person walking in it.
(2) Get video frames and predict the person walking in it.
(3) Apply Gait Recognition on a video and save the gait vector obtained into the database.
(4) Apply Gait Recognition on video frames and save the gait vector obtained into the database.
(5) Get frames from all the videos in a directory and save it at the same location as the video.
(6) Get frames from a video and save it at the same location as the video.

This application can be used as an assistive program to make the existing Surveillance System smarter (See Algorithm 1).

**Algorithm 1: Pose Estimation**

```
Input: Silhouette
Output: Pose Descriptor
Get_Frames_v5:; /* This module is used to
  get frames in the required specific shape
  and pixel size. We will be using YOLOv3
  (version 3), in particular YOLO trained on
  COCO data set. The functions implemented
  in this module are: */
isEmpty(path):; /* This function checks if
  the path specified is a valid directory. */
make_square(im, min_size, fill_colour):;
  /* This function fits an image into square
  frame. */
get_frames_and_save(path1):; /* This
  function takes a directory path as an input
  and creates frames for every video present
  in it. */
get_frames_and_save_from_video(path1):;
  /* This function takes a video path as an
  input and creates frames for
  the video. */
n parray_from_images(folder):; /* This
  function takes a directory path as an
  input. The folder should contain video
  frames in it. It returns a 4D numpy array
  that contains all the images in it. */
n parray_from_video(video_path):; /* Takes
  video path as an input and returns a 4D
  numpy array that contains all the video
  frames in the required specific format. */
human_pose_resnet(net, reuse = False,
  training = False):; /* Architecture of
  Part Detector network is used in Human Pose
  Estimation via Convolutional Part Heat Map
  Regression. */
human_pose_neural_network:; /* This module
  contains 3 classes that are used to get a 4D
  nparray of video frames and returns the
  spatial features extracted. We use a
  pre-trained model "Human3.6 m.ckpt",
  which is trained over a large data set
  "Human3.6 m". This model uses the module
  part_detector. */
gait_neural_network:; /* This module has 2
  classes in it. It uses multilayer
  recurrent cells and GRU to extract
  temporal features that are then aggregated
  with average temporal pooling into one-
  dimensional identification vector with
  good discriminatory properties. */
classGaitNN(object):; /* Responsibility of
  GaitNN is the further processing of the
  generated spatial features into one-
  dimensional pose descriptors with the use
  of a residual convolutional network. */
```

## 4.8 | Classification

This module deals with predicting a person in a new video. Using the database csv, we train a classification model. The new video is then processed and identification vector (converted into a python list) is obtained. This identification list is then passed to the model as test dataframe and the name of the person is predicted. The functions in this module are given in Algorithm 2.

**Algorithm 2: Classification**

```
predict_from_video(path_of_video,
path_to_csv = csv_path):; /* Takes path of a
  video as parameter: path_of_video, and
  path of the database csv as parameter:
  path_to_csv. path_to_csv has a default
  value that is defined in the module. This
  function uses previously mentioned
  function to get identification vector from
  the video and predicts the person (class),
  using the classification model. */
predict_from_frames(path_of_dir,
path_to_csv = csv_path):; /* The directory
  should contain video frames extracted
  using the program. path_to_csv has a
  default value that is defined in the module.
  This function uses previously mentioned
  function to get identification vector from
  the video frames and predicts the person
  (class), using the classification model. */
Gui_v2:; /* This module uses all the modules
  mentioned above and creates a graphical
  user interface that provides 6
  functionalities. The functions from other
  modules used are mentioned below: */
predict_from_video():; /* Get a video and
```

```
predict the person walking in it. */
predict_from_frames():; /* Get video frames
and predict the person walking in it. */
video_to_gait_and_save():; /* Apply Gait
  Recognition on a video and save the gait
  vector obtained into the database. */
frames_gait_implementation():; /* Get
  video frames and predict the person
  walking in it. */
get_frames_from_video():; /* Get frames
  from a video and save it at the same
  location as the video. */
get_frames_from_directory():; /* Get
  frames from all the videos in a directory
  and save it at the same location as the
  video. */
inception_resnet_v2:; [/* The major
  advancements in image recognition
  performance in recent years have been
  driven by deep convolutional networks
  [32]. The Inception architecture [33], for
  example, has been proven to offer
  excellent performance at a cheap
  computational cost. Inception v1 [34] is
  a 27-layer convolutional neural network
  (CNN). The Inception-ResNet-v2 [35]
  version of Inception is a more expensive
  hybrid version with much-enhanced
  recognition performance. */
complete_gait_implementation:; /* This
  module puts together all the 3
  implementations, namely: Getting Frames,
  Pose Estimation, and Gait Recognition. */
```

Temporal features are then extracted across these pose descriptors with the use of the multilayer recurrent cells - LSTM or GRU. All temporal features are finally aggregated with average temporal pooling into one-dimensional identification vector with good discriminatory properties.

### 4.9 | Results

This application uses the methodology of implementing pose estimation first and then gait recognition. These pose based representation tacitly capture biometric spatial features and temporal features more accurately than the usual GEI (Gait Energy Image). Due to this 2-step process, the processing is very robust against various covariate factors such as clothing, carrying conditions, shoe types and so on. The program is made in a way (in separate modules), that promotes adaptability. A simple GUI is implemented so that this application can be used by people with no technical or coding knowledge, as the GUI provides simple explanation and implements the working at the click of a button as depicted in Figure 9.

The best way of studying the results and the accuracy of the gait implementation is by evaluating the classification model in Table 1. For this, the data set is divided into two parts, 1 for training the classifier and other to test the classifier on. SVM classifier with Gaussian kernel was chosen because of the number of fields the data set has, that is, 1536 columns.

For getting proper result, the two data sets were created using 106 videos: train and test data set. The train data set contained gait values for 64 videos, and the test data set contained gait values for 42 videos. An SVM classification model was trained on the train data set, the test data set was used to test the accuracy of the classifier. The accuracy score was found to be 95.23%. The confusion matrix for the classification process is shown in Figure 10. Table 2 shows comparison of result with state-of-the-art.

## 5 | CONCLUSION AND FUTURE SCOPE

### 5.1 | Conclusion

A GUI-based application was made that runs gait recognition in the background. This application can be used as an assistive

**TABLE 1** Accuracy

| | 0 | 18 | 36 | 54 | 72 | 90 | 108 | 126 | 144 | 162 | 180 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 60.88 | 63.14 | 67.4 | 72.66 | 76.92 | 83.18 | 75.82 | 68.6 | 62.8 | 61.3 | 60.34 |
| **18** | 63.87 | 66.13 | 70.39 | 75.65 | 79.91 | 86.17 | 78.81 | 71.59 | 65.79 | 64.29 | 63.33 |
| **36** | 65.74 | 68 | 72.26 | 77.52 | 81.78 | 88.04 | 80.68 | 73.46 | 67.66 | 66.16 | 65.2 |
| **54** | 67.3 | 69.56 | 73.82 | 79.08 | 83.34 | 89.6 | 82.24 | 75.02 | 69.22 | 67.72 | 66.76 |
| **72** | 69.41 | 71.67 | 75.93 | 81.19 | 85.45 | 91.71 | 84.35 | 77.13 | 71.33 | 69.83 | 68.87 |
| **90** | 74.53 | 76.79 | 81.05 | 86.31 | 90.57 | 95.23 | 89.47 | 79.24 | 73.44 | 71.94 | 70.98 |
| **108** | 71.89 | 74.15 | 78.41 | 83.67 | 87.93 | 94.19 | 86.83 | 79.61 | 73.81 | 72.31 | 71.35 |
| **126** | 69.25 | 71.51 | 75.77 | 81.03 | 85.29 | 91.55 | 84.19 | 76.97 | 71.17 | 69.67 | 68.71 |
| **144** | 67.01 | 69.27 | 73.53 | 78.79 | 83.05 | 89.31 | 81.95 | 74.73 | 68.93 | 67.43 | 66.47 |
| **162** | 64.69 | 66.95 | 71.21 | 76.47 | 80.73 | 86.99 | 79.63 | 72.41 | 66.61 | 65.11 | 64.15 |
| **180** | 61.11 | 63.37 | 67.63 | 72.89 | 77.15 | 83.41 | 76.05 | 68.83 | 63.03 | 61.53 | 60.57 |

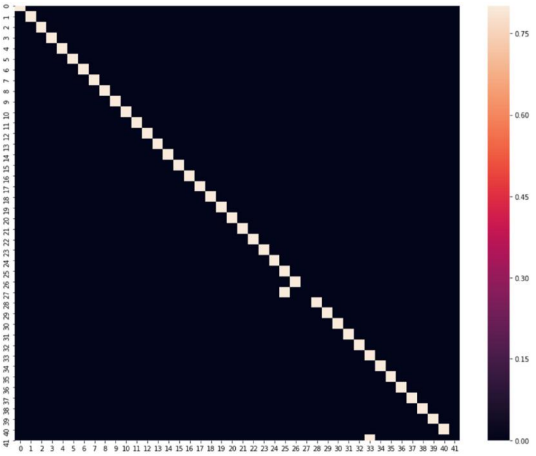


**FIGURE 10** Confusion matrix for support vector machine (SVM) model.

**TABLE 2** Comparison of results

| Cite | Mean accuracy |
|---|---|
| [36] | 73.49 |
| [37] | 49.72 |
| **Proposed** | **74.35** |

application that can assist an existing Surveillance System in an organisation. This application has a complex structure that uses the methodology of implementing frames extraction, pose estimation first and then gait recognition in the same order. All the neural networks used in the project are very robust and most of them are predefined models or architectures. 'Human3.6 m.ckpt', 'H3.6m-GRU-1.ckpt', and YOLOv3 are the predefined models used that are very accurate and fast. These pose based gait recognition tacitly capture biometric spatial features and temporal features more accurately than the usual GEI (Gait Energy Image). The spaciotemporal features are very robust to many factors that GEI cannot possibly achieve. Due to this 2-step process of implementing pose estimation before gait recognition, the processing is very robust against various covariate factors such as clothing, carrying conditions, shoe types and so on. The program is made in separate modules, that promotes adaptability. A simple GUI is implemented so that this application can be used by people with no technical or coding knowledge, as the GUI provides simple explanation and implements the working at the click of a button [38].

## 5.2 | Future scope

Creating a program or an application in modules is a great practise grouping related code into a module makes the code easier to understand and use. We can use any Python source file as a module by executing an import statement in some other Python source file. This enables the application or the software to be easy to alter and change. It also makes adding new functionalities into the program easier compared to any other software architecture. Thus, the software application has been created in modules, or packages, which opens new possibilities. These packages have separate functionalities and pass messages to each other. This means that if we want to change something in one package or implementation, we do not need to change anything in any other packages or how the program works as a whole. If any new functionality is to be added in the system, a new package can be created that would communicate with the other packages accordingly, and implement the new proposed functionality. The only thing that puts a constraint on this application is data gathering and processing capabilities.

Three likely future possibilities for this application are:

1. Making the whole smart surveillance work on a cloud. All the CCTV cameras will be needed to be connected through cloud too. The processing required can be handled through powerful cloud services like AWS, and the result can be obtained on handy devices.
2. If a powerful enough computer with high RAM, GPU, and CPU capabilities is available, the process of gait recognition and person prediction can be done in real-time. On a personal laptop, it takes around 2 min to do the whole process on a 15 s video. As the whole process goes through 4 neural, for a fast and efficient gait results, it would require a high-end Graphic processor.
3. Using a pretrained model for pose estimation and gait analysis is effective. But making a system with neural networks trained on personal video data set will help us creating a model specific to our data set.

## AUTHOR CONTRIBUTIONS

**Anubha Parashar**: Conceptualisation, Data curation, Formal analysis, Investigation, Methodology, Project administration, Resources, Software, Validation, Visualisation, Writing – original draft. **Apoorva Parashar**: Data curation, Investigation, Resources, Writing – review & editing. **Rajveer Shekhawat**: Project administration, Supervision, Validation, Writing – review & editing.

## ACKNOWLEDGEMENTS

No funding was provided for this work.

## CONFLICT OF INTEREST

The authors declare that they have no conflict of interests.

## DATA AVAILABILITY STATEMENT

Data available on request due to privacy/ethical restrictions

## ORCID

*Anubha Parashar* https://orcid.org/0000-0002-8474-3623

## REFERENCES

1. Rida, I., Almaadeed, S., Bouridane, A.: Gait recognition based on modified phase-only correlation. Signal, Image Video Process. 10(3), 463–470 (2016). https://doi.org/10.1007/s11760-015-0766-4

2. Rida, I., Almaadeed, N., Almaadeed, S.: Robust gait recognition: a comprehensive survey. IET Biom. 8(1), 14–28 (2019). https://doi.org/10.1049/iet-bmt.2018.5063
3. https://factly.in/of-all-the-cctv-cameras-available-to-the-police-in-india-64-in-telangana/
4. An, W., et al.: Improving gait recognition with 3d pose estimation. In: Chinese Conference on Biometric Recognition, pp. 137–147. Springer, Cham (2018)
5. Rida, I., et al.: A comprehensive overview of feature representation for biometric recognition. Multimed. Tool. Appl. 79(7), 4867–4890 (2020). https://doi.org/10.1007/s11042-018-6808-5
6. Rida, I., et al.: Improved human gait recognition. In: International Conference on Image Analysis and Processing, pp. 119–129. Springer, Cham (2015)
7. Rida, I., et al.: Robust model-free gait recognition by statistical dependency feature selection and globality-locality preserving projections. In: 2016 39th International Conference on Telecommunications and Signal Processing (TSP), pp. 652–655. IEEE (2016)
8. Rida, I., et al.: Improved model-free gait recognition based on human body part. In: Biometric Security and Privacy, pp. 141–161. Springer, Cham (2017)
9. Rida, I.: Towards Human Body-Part Learning for Model-free Gait Recognition (2019). arXiv preprint arXiv:1904.01620
10. Rida, I.: Temporal signals classification. (Classification de signaux temporels). Doctoral dissertation, Normandy University, France (2017)
11. Rida, I.: Feature Extraction for Temporal Signal Recognition: An Overview (2018). arXiv preprint arXiv:1812.01780
12. Liao, R., et al.: A model-based gait recognition method with body pose and human prior knowledge. Pattern Recogn. 98, 107069 (2020). https://doi.org/10.1016/j.patcog.2019.107069
13. Sokolova, A., Konushin, A.: Pose-based deep gait recognition. IET Biom. 8(2), 134–143 (2019). https://doi.org/10.1049/iet-bmt.2018.5046
14. Tavares, H.L., et al.: Tracking and re-identification of people using soft-biometrics. In: 2019 XV Workshop de Visão Computacional (WVC), pp. 78–83. IEEE (2019)
15. Luo, J., Tjahjadi, T.: Gait recognition and understanding based on hierarchical temporal memory using 3D gait semantic folding. Sensors 20(6), 1646 (2020). https://doi.org/10.3390/s20061646
16. Hasan, M.M., Mustafa, H.A.: Multi-level feature fusion for robust pose-based gait recognition using RNN. Int. J. Comput. Sci. Inf. Secur. 18(1) (2020)
17. Sheng, W., Li, X.: Siamese denoising autoencoders for joints trajectories reconstruction and robust gait recognition. Neurocomputing 395, 86–94 (2020). https://doi.org/10.1016/j.neucom.2020.01.098
18. Li, Na, Zhao, X., Ma, C.: A Model-Based Gait Recognition Method Based on Gait Graph Convolutional Networks and Joints Relationship Pyramid Mapping (2020)
19. Jun, K., et al.: Feature extraction using an RNN autoencoder for skeleton-based abnormal gait recognition. IEEE Access 8, 19196–19207 (2020). https://doi.org/10.1109/access.2020.2967845
20. Zou, Z., et al.: Object Detection in 20 Years: A Survey (2019). arXiv preprint arXiv:1905.05055
21. Fang, H.S., et al.: Rmpe: regional multi-person pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2334–2343 (2017)
22. Zhang, R., Vogler, C., Metaxas, D.: Human gait recognition. In: 2004 Conference on Computer Vision and Pattern Recognition Workshop. 18. IEEE (2004)
23. Chen, C., et al.: R-CNN for small object detection. In: Asian Conference on Computer Vision, pp. 214–230. Springer, Cham (2016)
24. Girshick, R.: Fast r-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)
25. Ren, S., et al.: Faster r-CNN: towards real-time object detection with region proposal networks. Adv. Neural Inf. Process. Syst. 28 (2015)
26. Liu, W., et al.: Ssd: Single shot multibox detector. In: European Conference on Computer Vision, pp. 21–37. Springer, Cham (2016)
27. Redmon, J., et al.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
28. Redmon, J., Ali, F.: Yolov3: An Incremental Improvement (2018). arXiv preprint arXiv:1804.02767
29. Lin, T.-Yi, et al.: Microsoft coco: common objects in context. In: European Conference on Computer Vision, pp. 740–755. SpringerCham (2014)
30. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. 25 (2012)
31. Yue-Hei Ng, J., et al.: Beyond short snippets: deep networks for video classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4694–4702 (2015)
32. Simonyan, K., and Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
33. Szegedy, C., et al.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
34. Szegedy, C., et al.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: Thirty-first AAAI Conference on Artificial Intelligence (2017)
35. Szegedy, C., et al.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
36. Wu, Z., et al.: A comprehensive study on cross-view gait based human identification with deep cnns. IEEE Trans. Pattern Anal. Mach. Intell. 39(2), 209–226 (2016). https://doi.org/10.1109/tpami.2016.2545669
37. Liao, R., et al.: A model-based gait recognition method with body pose and human prior knowledge. Pattern Recogn. 98, 107069 (2020). https://doi.org/10.1016/j.patcog.2019.107069
38. Parashar, A., et al.: Intra-class Variations with Deep Learning-Based Gait Analysis: A Comprehensive Survey of Covariates and Methods. Neurocomputing (2022)


**How to cite this article:** Parashar, A., Parashar, A., Shekhawat, R.S.: A robust covariate-invariant gait recognition based on pose features. IET Biome. 11(6), 601–613 (2022). https://doi.org/10.1049/bme2.12103