scientific reports

OPEN

Enhancing gait recognition by multimodal fusion of mobilenetv1 and xception features via PCA for OaA-SVM classification

Akash Pundir[1,2], Manmohan Sharma[1], Ankita Pundir[3], Dipen Saini[1], Khmaies Ouahada[4], Salil bharany[5]✉, Ateeq Ur Rehman[6]✉ & Habib Hamam[4,7,8,9]

Gait recognition has become an increasingly promising area of research in the search for noninvasive and effective methods of person identification. Its potential applications in security systems and medical diagnosis make it an exciting field with wide-ranging implications. However, precisely recognizing and assessing gait patterns is difficult, particularly in changing situations or from multiple perspectives. In this study, we utilized the widely used CASIA-B dataset to observe the performance of our proposed gait recognition model, with the aim of addressing some of the existing limitations in this field. Fifty individuals are randomly selected from the dataset, and the resulting data are split evenly for training and testing purposes. We begin by excerpting features from gait photos using two well-known deep learning networks, MobileNetV1 and Xception. We then combined these features and reduced their dimensionality via principal component analysis (PCA) to improve the model's performance. We subsequently assessed the model using two distinct classifiers: a random forest and a one against all support vector machine (OaA-SVM). The findings indicate that the OaA-SVM classifier manifests superior performance compared to the others, with a mean accuracy of 98.77% over eleven different viewing angles. This study is conducive to the development of effective gait recognition algorithms that can be applied to heighten people's security and promote their well-being.

Keywords Human identification, Gait, Deep learning, Biometric, Pretrained models, Security, Healthcare

Since encouraging results in distinguishing and confirming humans without any direct physical interaction with machines have been demonstrated, human gait identification has been a focus of attention. This method relies upon the use of a range of sensors, including cameras, accelerometers, and gyroscopes, to record an individual's gait[1–4]. The distinctive characteristics of the human gait, including stride length, walking speed, and body position, make it a feasible choice for use as a biometric feature for identification[5]. One of their main advantages is that gait biometrics are more impregnable than other biometrics because they are difficult to retroflex or falsify. Every person has an involuntary, natural gait that reflects their physical characteristics, such as height, weight, and leg length, as well as their psychological characteristics, such as personality and mood. For these reasons, facial recognition technology is a valuable tool for detecting individuals who may effort to conceal their genuine identity[6]. Advanced developments in computer vision and machine learning have led to the creation of numerous strategies for gait recognition. Typically, these methods start by evoking features from gait data and then classify

[1]School of Computer Science and Engineering, Lovely Professional University, Phagwara, India. [2]Department of Computer Science and Engineering, Dr B.R. Ambedkar National Institute of Technology, Jalandhar, India. [3]School of Bioengineering and Biosciences, Lovely Professional University, Phagwara, India. [4]Department of Electrical and Electronic Engineering Science, School of Electrical Engineering, University of Johannesburg, Johannesburg 2006, South Africa. [5]Institute of Engineering and Technology, Chitkara University, Chitkara University, Punjab, India. [6]School of Computing, Gachon University, Seongnam 13120, Republic of Korea. [7]Faculty of Engineering, Uni de Moncton, Moncton, NB E1A3E9, Canada. [8]Hodmas University College, Taleh Area, Mogadishu, Somalia. [9]Bridges for Academic Excellence, Tunis, Centre Ville, Tunisia. ✉email: salil.bharany@gmail.com; 202411144@gachon.ac.kr

the data using a machine learning algorithm[7]. Although deep learning models have been successful in many computer vision applications, it has become normal practice to use them for feature extraction. Gait identification has greatly advanced in recent decades. The demand for trustworthy biometric solutions has fuelled this advancement. In many real-world scenarios, traditional biometric techniques such as fingerprint and iris identification[8,9] have drawbacks. The fact that these techniques are not always precise is one major problem. However, without necessitating direct physical contact, gait recognition offers a scalable and noninvasive alternative. Access control, surveillance, and forensic investigation are just a few of the applications for this technology.

Challenges in gait recognition

Gait recognition has considerable potential, but before it can be a useful solution, there are still several obstacles to overcome. A significant obstacle is the substantial intraclass variability, or variations in a person's stride resulting from their physical condition, clothing, and environment. Low interclass variability, or difficulty in distinguishing between individuals due to their identical gait patterns, is another problem. This may result in incorrect categorizations, which could be problematic[10].

To address these challenges, researchers have explored various machine learning techniques[11–14]. These methods aim to accurately capture the unique features of human gait. In this area, deep learning algorithms have performed very well in recent years, achieving the best results on many available benchmark datasets. This achievement has been achieved on several different benchmark datasets. Unfortunately, these methods typically require a large amount of labeled data as well as significant processing resources, which renders them unsuitable for use in real-world applications. In light of this, the intention of the current research is to overcome these issues by putting forward the idea of a practical gait identification system that makes use of a mixture of deep learning models and machine learning algorithms.

Major contributions

The major outcomes of this research are as follows:

1. Development of a novel gait recognition system: This research presents a new approach to gait recognition by combining MobileNetV1 and Xception deep learning models for feature extraction, followed by feature fusion, dimensionality reduction using PCA, and classification using OaA-SVM. To the best of our knowledge, this combination has not been used before for gait identification and has achieved high accuracy.
2. High Accuracy Results: This new method for gait recognition achieved much better results than did the current systems, reaching an average accuracy of 98.77%. This high accuracy is because of the effective combination of feature fusion, reducing the size of the data, using deep learning models, and classification techniques. These issues were improved through thorough testing and validation.

Manuscript organization

The manscuript is structured in the following manner: Section "Related work" provides a concise overview of the research conducted by various scholars in tabular format. Section "Methodology" provides an overview of the proposed method, including the dataset utilized, the data preprocessing techniques, the feature extraction methods, the feature fusion approaches, the feature reduction strategies, and the classification algorithms. Section "Results and discussion" provides an evaluation of the outcomes achieved using the proposed approach, while Section "Conclusion" provides the final remarks and summary of the study.

Related work

The difficulty of recognizing persons from films shot under different situations has been addressed by numerous methodologies. In recent years, video-based gait identification has become a major research topic. This section covers the proposed cutting edge deep learning techniques based on gait recognition.

Gait recognition using pretrained models

Researchers used local and global filter information to improve video frames in[15]. Data augmentation was then used to expand the dataset. After fine-tuning the two pretrained models, ShuffleNet and MobileNetV2, features were taken, and the resultant features were merged using a serial technique.

With the use of the equilibrium state optimization controlled Newton–Raphson (ESOnCR) algorithm, the features are enhanced from fused features. Finally, on the CASIA-B database, classification using the quadratic support vector machine (QSVM) produced a mean accuracy of 95.19% across 11 angles[16]. At an angle of 180, the model's accuracy was at its highest point—99.8%.

Similarly, in[17], two similar pretrained models, EfficientNet-B0, were used for feature extraction; one of them was trained using motion regions based on optical flow, and the other model was separately trained on extracted frames that were enhanced. For selecting the hyperparameters, Bayesian optimization was used, following which features were extracted from both models and fused with the use of the Sq-Parallel Fusion (SqPF) approach, and further features were improved with the use of the entropy controlled tiger optimization (EcTO) algorithm. Classification was performed with an extreme learning machine (ELM) classifier, and the model achieved mean accuracies of 92.04% and 94.97% on CASIA-B and CASIA-C, respectively[16].

In a different approach described in[18], two pretrained models for feature extraction, VGG19 and AlexNet, were utilized. The features extracted from both models were fused, and the best features were selected with the help of fuzzy entropy-controlled skewness (FEcS). For the purpose of classification, random forest[19] was used, and it achieved a mean accuracy of 93.3% on the CASIA-B dataset across 11 angles[16].

In[20], ResNet101 deep ConvNet was used for feature extraction; the features were taken from both the global mean pooling layer and the fully connected layer of the model, selected with the help of the kurtosis controlled entropy (KcE) approach and further fused with the help of the correlation approach. OaA-SVM achieved an accuracy of 95.26% and 96.6% on the CASIA-B dataset and the user's own dataset, respectively. The method proposed by[12] involves selecting and fine-tuning two pretrained deep models, extracting features from VGG19 and MobileNetV2, fusing features from these two models using discriminant correlation analysis (DCA), optimizing features with the use of a modified Moth-Flame optimization algorithm (MMFO), and for the purpose of classification, using an extreme learning machine classifier. The proposed method achieves mean accuracies of 91.20% and 98.60% on the CASIA-B and TUM GAID datasets, respectively[21].

In a different approach[22], the researchers used a single pretrained model for feature extraction, named VGG16. The dimension of the feature is reduced with the use of an approach based on principal score-based kurtosis (PSbK). Since only one model was used for feature extraction, there was no feature fusion step. The classification is performed with the help of the OaA-SVM classifier. On the CASIA-B dataset[16], or only for the first six angles from 0 to 90, the model achieved a mean accuracy of 95.83%.

In another study based on pretrained models[23], researchers used ResNet101 and InceptionV3 for extracting features; then, the features were optimized based on a nature-inspired algorithm known as improved ant colony optimization (IACO). For evaluation, three angles were considered from the CASIA-B dataset[16], namely, 0, 18, and 180. The classification using the cubic SVM yielded a mean accuracy of 95.7% for these 3 angles.

In research by[24], researchers used Inception-ResNetV2 and NASNet Mobile for extracting features. The extracted features were optimized with another nature-inspired algorithm known as the modified whale optimization algorithm (MWOA). After optimization, the features were fused via the MDeSF (mean absolute deviation extended serial fusion) approach. The experiment was performed on all angles of the CASIA-B dataset. The classification using Cubic SVM (C-SVM) achieved a mean accuracy of 89%.

In[25], the authors used DenseNet201 for extracting features from the extracted video frames. The features were reduced using the firefly algorithm (FA) and another approach based on the skewness approach (SA). The reduced feature vectors were fused via serial fusion. The experiment was conducted on 3 angles of the CASIA-B dataset: 18, 36, and 54. The classification using OaA-M-SVM achieved a mean accuracy of 94.26%.

In Table 1, the methods discussed above are summarized in a crisp manner by breaking down the complete approach. These works are based entirely on pretrained models; furthermore, there are many other studies on gait based on different deep learning methods, such as autoencoders, generative adversarial networks, recurrent neural networks, capsule networks, and simple convnets; the work on these methods is discussed in this research paper[26].

After reviewing the literature, this study is motivated by the need for robust and noninvasive person identification methods. In our model design, we strategically address the observed limitations in existing studies to improve the effectiveness of gait identification. By leveraging the advantages of MobileNetV1 and Xception for feature extraction, we aim to capture intricate gait patterns effectively. The fusion of features from these two deep learning networks enriches the model's ability to discern complex characteristics. Additionally, the application of principal component analysis (PCA) reduces dimensionality. The utilization of the OaA-SVM classifier yields an impressive mean accuracy of 98.77% across eleven viewing angles, surpassing reported accuracies from various existing models. This design decision is grounded in the aim of achieving superior gait recognition results.

Methodology

The proposed method employs a deep learning-based approach to identify humans. Figure 1 depicts the flow of the approach. The approach comprises extracting frames from videos, preprocessing the extracted frames, using transfer learning for feature mining, concatenating the extracted features, reducing the extracted features, and then classifying the frames. In the following section, these steps are discussed in detail.

Dataset

In January 2005, CASIA-B[16], a large multiview gait database, was developed. It covers the gait data of 124 people from 11 different perspectives. The dataset is well known in the area of gait research and takes into consideration the angle from which a person walks, the clothes that a person wears, and the object a person is carrying. In

Reference	Year	Feature Extraction	Feature Selection/Improvement	Feature Fusion	Dataset	No. of Angles	Classifier	Accuracy%
[15]	2023	MobileNetV2 and ShuffleNet	ESOcNR	Serial Fusion	CASIA-B	11	Q-SVM	95.19
[17]	2023	Dual EfficientNet- B0	EcTO	SqPF	CASIA-B	11	ELM	92.04
[18]	2022	VGG19 and AlexNet	FEcS	Parallel Fusion	CASIA-B	11	RF	93.3
[20]	2022	ResNet101	KcE	-	CASIA-B	11	OaA-SVM	95.26
[27]	2022	VGG19 and MobileNetV2	MMFO	DCA	CASIA-B	11	ELM	91.2
[22]	2022	VGG16	PSbK	–	CASIA-B	6 (0–90)	OaA-SVM	95.83
[23]	2022	ResNet101 and InceptionV3	IACO	–	CASIA-B	3 (0,18,180)	C-SVM	95.7
[24]	2021	Inception-ResNetV2 and NASNet Mobile	MWOA	MDeSF	CASIA-B	11	C-SVM	89
[25]	2020	DenseNet201	FA and SbA	Serial Fusion	CASIA-B	3(18,36,54)	OaA -M- SVM	94.26

Table 1. Deep learning approaches based on pretrained models.

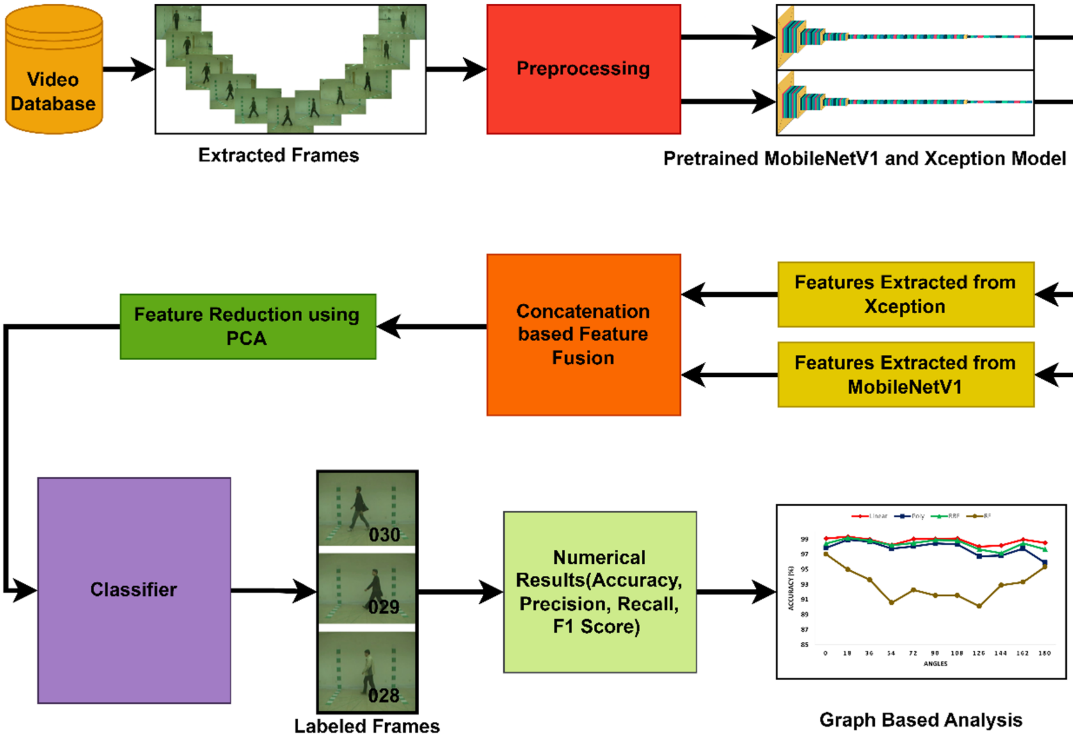
Video Database
Extracted Frames
Preprocessing
Pretrained MobileNetV1 and Xception Model
Features Extracted from Xception
Features Extracted from MobileNetV1
Concatenation based Feature Fusion
Feature Reduction using PCA
Classifier
030
029
028
Labeled Frames
Numerical Results(Accuracy, Precision, Recall, F1 Score)
ACCURACY (%)
ANGLES
Graph Based Analysis

Figure 1. Architecture of the proposed methodology.

addition to visual data, it displays individual silhouettes extracted from video formats. The data contain 3 variations of the person walking: walking with a bag, walking in a coat, and walking normally. The dataset contains this variation from eleven distinct viewing angles from 0 to 180.

In our research, frames whose original size was 240 × 320 were extracted from the videos. These frames were resized to 224 × 224. As shown in Fig. 2, the dataset was split 50–50 for training and testing, employing a holdout validation technique. This ratio of splitting was chosen due to the structure of the dataset. Anglewise analysis was performed for all 11 angles in the dataset. As shown in Fig. 3, colorful frames were used for research, and

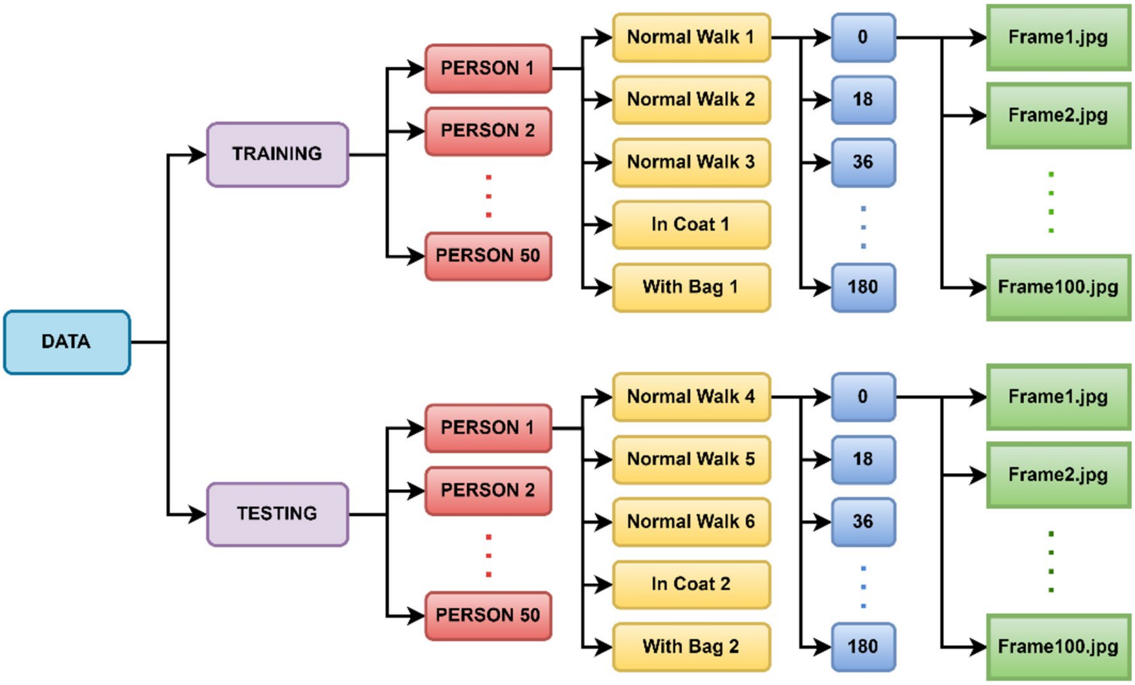

Figure 2. Directory structure of the dataset after splitting.

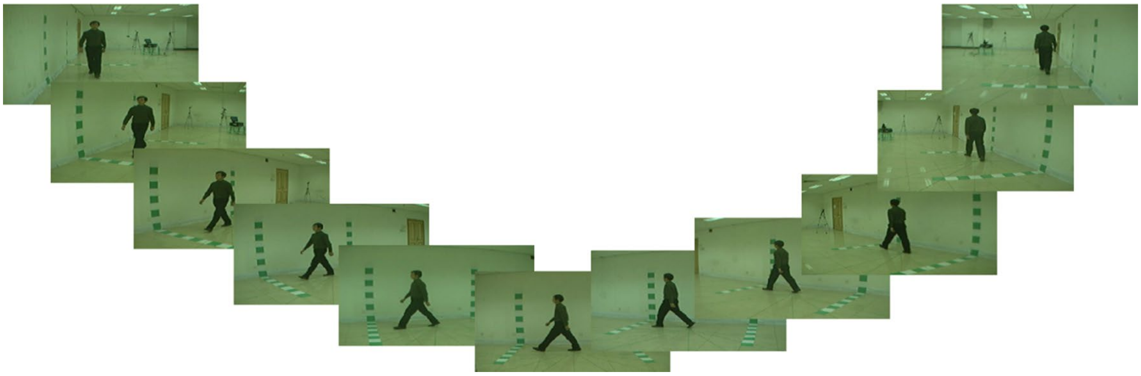

Figure 3. Sample frames extracted from the CASIA-B database.

empty frames were discarded during extraction. Table 2 provides details of the modified dataset used from the CASIA-B dataset[16].

Normalization

The normalization of pixel values is the initial stage in our process. Normalization is a technique that is commonly used in data preprocessing. Normalization is the process of rescaling a variable's values to a common scale so that they may be compared directly. Pixel normalization is an important preprocessing step in image processing that guarantees that pixel values are within a given range. Pixel normalization ensures that the model receives inputs with a consistent scale and distribution. In an image, pixel standards can range from 0 to 255, where 0 means total darkness and 255 means total brightness. We normalize these pixel values to be between 0 and 1. This helps avoid numerical overflow and underflow and makes data processing easier. Additionally, normalizing the image improves the contrast between pixels, making details that were hard to see in the original image more visible. Normalization can be represented mathematically as follows:

$$\frac{x - x_{min}}{x_{max} - x_{min}} = x_{normalized} \tag{1}$$

If x is the original pixel value, x_{min} is the picture's lowest pixel amount, and x_{max} is the picture's maximum pixel value. The resulting $x_{normalized}$ value is in the [0,1] range.

The rescale function is commonly employed in image processing libraries such as OpenCV, Scikit-Image, or Keras for the purpose of achieving normalization. The ImageDataGenerator class from Keras is utilized in this study to standardize the pixel values. The rescale parameter of the ImageDataGenerator class is assigned a value of 1.0/255.0, which scales down the pixel values to a range between 0 and 1. Normalization is an essential preprocessing step since it enhances the model's ability to efficiently process input data. It improves the precision of the model and reduces numerical inaccuracies caused by fluctuations in pixel values.

Feature extraction

The next step is to use pretrained convolutional neural network (CNN) models to obtain attributes from these images. CNNs are good at automatically understanding and obtaining features from raw pixel data, which makes them useful for feature extraction tasks in image processing. Traditional neural networks require considerable computational power and memory to train because every neuron in one layer is linked to every neuron in the earlier layer. CNNs, on the other hand, use convolutional layers, which reduce the number of parameters and calculations needed for training by merely linking a subset of neurons from the earlier layer to a tiny receptive field in the current layer. In the next subsection, CNNs are discussed along with various layers of CNNs.

Convolutional neural network

ConvNet is a type of neural network that is remarkably suitable for picture processing tasks. A typical CNN will have the following layers: an input node, many levels of convolutional processing, another type of layer known as a pooling layer, and finally, a fully connected layer.

Convolutional layer. Sliding a kernel (also called a filter or a pattern detector) across the input picture, multiplying the kernel by the given input pixels, and then summing the results is the convolution process. For a

No. of person	Frame format	Frame size	No. of angles	No. of variations
50	'JPG'	224 × 224	11	3

Table 2. Subset of the CASIA-B dataset used in this research.

convolutional layer, the resultant feature map O is acquired by directing a convolution operation $\star$ between the entered image I and the filter F, followed by an optional bias term b and an activation function σ:

$$O = \sigma(I \star F + b) \tag{2}$$

The convolution operation $\star$ is defined as calculating the elementwise product and total at each place while moving the filter over the given image:

$$I \star F = \sum\nolimits_{i=1}^{F_h} \sum\nolimits_{j=1}^{F_w} \sum\nolimits_{k=1}^{C} I_{m+i-1,n+j-1,k} F_{i,j,k} \tag{3}$$

where F_h and F_w are the height and width of the filter, respectively, and C is the number of channels in the input picture. The resultant feature map has a height and width given by:

$$O_h = \frac{I_h + 2P_h - F_h}{S_h} + 1 \tag{4}$$

$$O_w = \frac{I_w + 2P_w - F_w}{S_w} + 1 \tag{5}$$

where I_h and I_w are the height and width of the input image, respectively; P_h and P_w are the padding values along each dimension; and S_h and S_w are the stride values along each dimension.

Pooling layer. The pooling operation is used to shrink the proportions of the attribute maps and to introduce some degree of translation invariance to the output.

For a pooling layer, the resultant attribute map O is acquired by applying a pooling function $\mathcal{P}$ over nonoverlapping regions of size $F_h \times F_w$ in the input feature map I:

$$O_{m,n} = \mathcal{P}\left(I_{mS_h:mS_h+F_h,nS_w:nS_w+F_w}\right) \tag{6}$$

where $\mathcal{P}$ can be either max pooling or average pooling:

$$\mathcal{P}\text{mx}(X) = \max(X) \tag{7}$$

$$\mathcal{P}\text{ag}(X) = \frac{1}{|X|} \sum (X) \tag{8}$$

The height and width of the output feature map are given by:

$$O_h = \frac{I_h - F_h}{S_h} + 1 \tag{9}$$

$$O_w = \frac{I_w - F_w}{S_w} + 1 \tag{10}$$

where I_h and I_w are the dimensions of the input feature map, and S_h and S_w are the stride values along each dimension.

Batch normalization layer. The inputs to a neural network layer may be normalized through a method called batch normalization (BN). The normalization of inputs to a layer is achieved by removing the layer's average and dividing by its standard deviation. The output of this layer is calculated as follows:

$$F_k = \frac{F_k - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}} \cdot \gamma_k + \beta_k \tag{11}$$

where F_k is the activation of feature map k, μ_k and σ_k are the mean and standard deviation of F_k over the batch, ϵ is a small constant to prevent division by zero, and γ_k and β_k are learned scaling and shifting parameters.

Fully connected (FC) layer. This layer is a type of layer in ConvNets where each neuron in the current layer is bonded to every other neuron in the next layer. For a fully connected layer, the output vector O is obtained by applying a linear transformation to the input vector I, followed by an optional bias term b and an activation function σ:

$$O = \sigma(WI + b) \tag{12}$$

where W is a weight matrix that has a shape of (O_d, I_d), O_d is the output dimension and I_d is the input dimension.

Pretrained models

In machine learning and deep learning, the concept of transfer learning is utilized to acquire the skills while solving one issue and applying them to another similar problem. The idea behind transfer learning is to use a model trained on a large, complex dataset as a base for a similar problem with a smaller dataset. By using these pretrained models, we can save the time and computational resources needed to train a model from the beginning. Instead, we can use the features already learned by the model and use it to extract features for our specific task. This approach is especially helpful when we have limited data or limited computational resources.

MobileNetV1[28] and Xception[29] are two pretrained CNN models that are competent on the ImageNet dataset. MobileNetV1 is a light CNN model devised for mobile devices with fewer parameters than other CNN models. Xception, on the other hand, is a more complex CNN model that has shown good performance in different picture recognition tasks.

In this research, pretrained MobileNet and Xception models are used. The fully connected layers are excluded from the model. This allows features to be extracted from the intermediate layers of the CNN models instead of using them for classification tasks. In Table 3, the architecture of MobileNetV1 used in the research is given.

The Xception model is then expanded to include a global average pooling layer. In a pooling procedure, global average pooling calculates the average of all feature maps in the last convolutional layer. This process compresses the spatial dimensions of the feature maps and yields a vector that encapsulates the key features of the primary picture. As one of the features of the Xception model, the output of the global average pooling layer is taken. In Table 4, the architecture of the Xception model is given, including the different layers, their output sizes and the number of filters.

New models for MobileNet and Xception are created by stating the input and output layers. All the layers of the MobileNet and Xception models are set to be nontrainable by fixing the "trainable" parameter to "False". This guarantees that the pretrained weights of the CNN models are not modified during feature extraction.

In our feature extraction process, the global pooling layers of both the MobileNet and Xception models consistently produced outputs with shapes of (1024) and (2048), respectively, across various viewing angles. This uniformity is intentional and aligns with the chosen architectures, reflecting the condensed representation of features by the pooling layers, ensuring consistent input dimensions for subsequent layers in the gait recognition model. The consistent output shapes contribute to the stability and generalization capability of the overall network.

The feature vectors are then obtained from the input images via the MobileNet and Xception models. The "predict" method of the Keras library is employed to obtain the characteristics from the middle layers of the CNN models. The output of the "predict" method is a matrix of feature vectors that represents the important features of the input images. Table 5 summarizes the sizes of the features extracted from MobileNetV1 and Xception across various viewing angles, while Fig. 4 shows the numbers of samples across various viewing angles, which are the same for both the MobileNet and Xception models. Figure 5 shows the time taken by both models for feature extraction for their training and testing datasets across various viewing angles.

Finally, the feature matrices are reshaped into a 2D array where every row represents a single image and every column represents a feature. This creates a feature matrix that can be used as input to a machine learning model for training and testing.

Layer name	Type of layer	Output size	Number of filters
input_1	Input	224 × 224 × 3	–
conv1	Convolutional	112 × 112 × 32	32
conv_dw_1	Depthwise separable convolutional	112 × 112 × 32	–
conv_pw_1	Pointwise convolutional	112 × 112 × 64	64
conv_dw_2	Depthwise separable convolutional	56 × 56 × 64	–
conv_pw_2	Pointwise convolutional	56 × 56 × 128	128
conv_dw_3	Depthwise separable convolutional	28 × 28 × 128	–
conv_pw_3	Pointwise convolutional	28 × 28 × 128	128
conv_dw_4	Depthwise separable convolutional	28 × 28 × 256	–
conv_pw_4	Pointwise convolutional	28 × 28 × 256	256
conv_dw_5	Depthwise separable convolutional	14 × 14 × 256	-
conv_pw_5	Pointwise convolutional	14 × 14 × 512	512
conv_dw_6	Depthwise separable convolutional	14 × 14 × 512	-
conv_pw_6	Pointwise convolutional	14 × 14 × 512	512
conv_dw_7	Depthwise separable convolutional	14 × 14 × 512	-
conv_pw_7	Pointwise convolutional	14 × 14 × 512	512
conv_dw_8	Depthwise separable convolutional	14 × 14 × 512	–
conv_pw_8	Pointwise convolutional	14 × 14 × 512	512
conv_dw_9	Depthwise separable convolutional	7 × 7 × 512	–
conv_pw_9	Pointwise convolutional	7 × 7 × 1024	1024
avg_pool	Global average pooling	1 × 1 × 1024	–

Table 3. Architecture/structure of the MobileNetV1 model.

Layer name	Type of layer	Output size	Number of filters
input_1	Input	299 × 299 × 3	–
block1_conv1	Separable convolutional	149 × 149 × 32	32
block1_conv2	Separable convolutional	147 × 147 × 64	64
block2_sepconv1	Separable convolutional	147 × 147 × 128	–
block2_sepconv2	Separable convolutional	147 × 147 × 128	128
block3_sepconv1	Separable convolutional	73 × 73 × 256	-
block3_sepconv2	Separable Convolutional	73 × 73 × 256	256
block4_sepconv1	Separable convolutional	37 × 37 × 728	-
block4_sepconv2	Separable convolutional	37 × 37 × 728	728
block5_sepconv1	Separable Convolutional	19 × 19 × 728	-
block5_sepconv2	Separable convolutional	19 × 19 × 728	728
block5_sepconv3	Separable convolutional	19 × 19 × 728	728
block6_sepconv1	Separable convolutional	10 × 10 × 728	–
block6_sepconv2	Separable convolutional	10 × 10 × 1024	1024
block6_sepconv3	Separable convolutional	10 × 10 × 1024	1024
avg_pool	Global average pooling	1 × 1 × 1024	–

Table 4. Architecture of the Xception model.

Configuration	Size of MobileNet features	Size of Xception features	Size of fused feature vector
For 000 degrees	(25,840, 1024)	(25,840, 2048)	3072
For 018 degrees	(27,719, 1024)	(27,719, 2048)	3072
For 036 degrees	(26,720, 1024)	(26,720, 2048)	3072
For 054 degrees	(23,041, 1024)	(23,041, 2048)	3072
For 072 degrees	(15,984, 1024)	(15,984, 2048)	3072
For 090 degrees	(14,812, 1024)	(14,812, 2048)	3072
For 108 degrees	(15,468, 1024)	(15,468, 2048)	3072
For 126 degrees	(19,330, 1024)	(19,330, 2048)	3072
For 144 degrees	(19,986, 1024)	(19,986, 2048)	3072
For 162 degrees	(20,050, 1024)	(20,050, 2048)	3072
For 180 degrees	(20,792, 1024)	(20,792, 2048)	3072

Table 5. Features extracted from MobileNetV1 and Xception for various viewing angles.

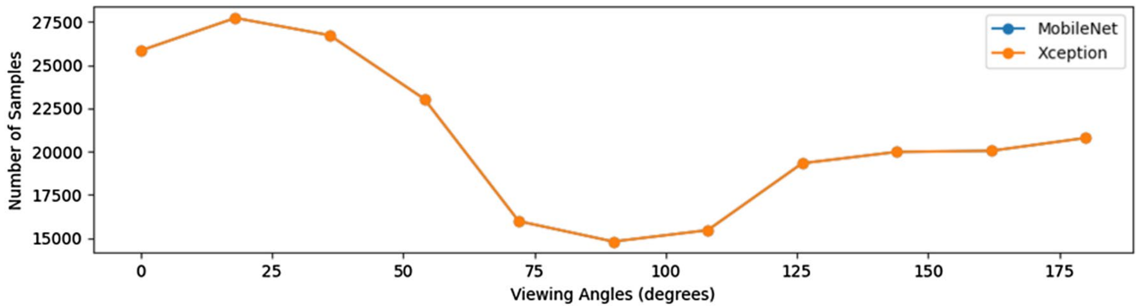

Figure 4. Number of Samples for MobileNetV1 and Xception.

Feature fusion

The next step after extracting features from the two pretrained models is to fuse these feature vectors. A method for combining the characteristics retrieved by various neural networks or layers is called feature fusion. Using the distinct features that are recorded by each network or layer, feature fusion seeks to improve the model's output. In our study, we use concatenation to implement feature fusion to enhance the accuracy of the picture classification model.

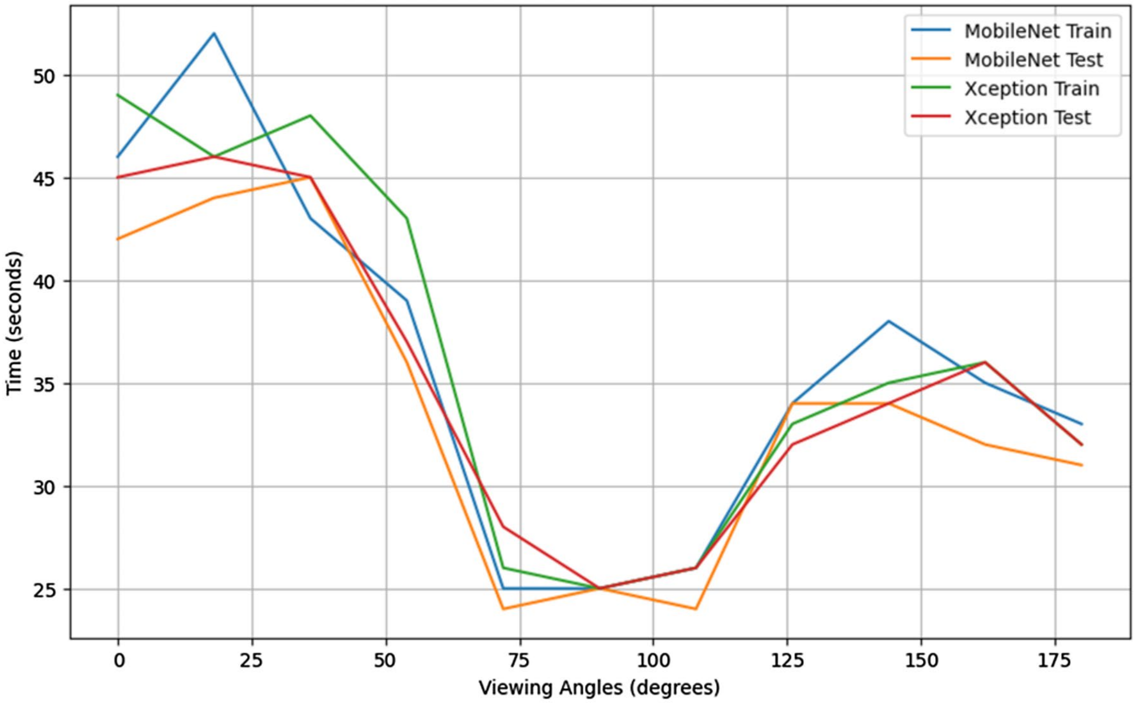

Figure 5. Time Taking for Feature Extraction at Different Angles.

Combining two or more tensors along a designated axis is known as concatenation. All of the information from the input tensors is contained in the greater dimensionality resultant tensor. If the tensor shapes are consistent along the designated axis, concatenation can be applied to tensors of various shapes.

Let us assume that the feature set generated by MobileNet has dimensionality M and that the feature set generated by Xception has dimensionality N. When two feature sets are concatenated, a new feature set with dimensionality $M + N$ is created.

The concatenation operation is mathematically represented as follows:

Let X be the feature set generated by MobileNet with dimensionality M, and let Y be the feature set generated by Xception with dimensionality N. X and Y can be concatenated along the second axis (axis 1) to obtain a new feature set Z with dimensionality $M + N$:

$$Z = [X, Y] \tag{13}$$

where $[X, Y]$ represents the concatenation of X and Y along axis 1.

The MobileNet and Xception models can extract features from the input images because they have already been trained on a sizable image dataset. The features that these models have extracted are complementary to one another because they were trained on different parts of the image data. Through feature concatenation, the performance of each model's unique features may be utilized to enhance the picture classification model. The feature tensors extracted from the MobileNet and Xception models have distinct dimensions. Thus, it is necessary to ensure that the features have the same dimensionality before concatenating them. Furthermore, our method involves reshaping the feature tensors from both models into a 1D vector using the reshape function. The size of the postfusion feature vector is 3072. This dimensionality reflects the combined information extracted through the fusion process, contributing to the comprehensive representation of gait patterns in our model.

Feature selection

The next step after fusing the extracted features is to reduce the dimensionality of the feature vector, and feature selection via principal component analysis (PCA)[30] is a common practice. The entirety of the data's natural variability as much as possible is preserved by PCA's dimensionality reduction. Given a dataset X of n samples, each having p features, the principal components are linear combinations of the initial features, which are generated through principal component analysis (PCA) to produce a new set of p orthogonal features known as principal components. The variation in the first principal component is the most significant, followed by the variation in the second principal component, which is the second most significant, and so forth.

Figure 6 displays the cumulative explained variance of the principal components for our feature vector. The x-axis represents the number of major components, while the y-axis represents the cumulative explained variance up to that specific number of components. This graph is valuable for selecting the optimal amount of main components to retain for subsequent investigation.

PCA computes the covariance matrix C:

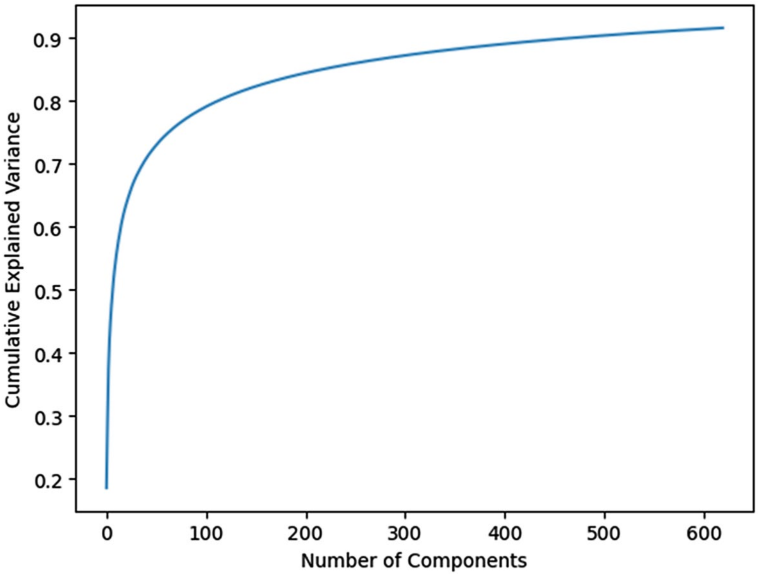

Figure 6. Cumulative explained variance for different numbers of components.

$$C = \frac{1}{n}\left(X - \overline{X}\right)^T\left(X - \overline{X}\right) \tag{14}$$

where $\overline{X}$ is the mean of dataset X and n is the number of samples in X.

The next step in principal component analysis (PCA) involves computing the eigenvectors and eigenvalues of the covariance matrix C. The eigenvectors illustrate the directions that contain the greatest amount of variation in the dataset, and the eigenvalues that correlate to those eigenvectors illustrate the amount of variance that may be found along those routes.

The eigenvectors and eigenvalues can be computed using the following equation:

$$Cv = \lambda v \tag{15}$$

where v is the eigenvector and λ is the eigenvalue.

Finally, PCA selects the top k eigenvectors (those with the highest eigenvalues) and projects the dataset onto these eigenvectors to obtain a lower-dimensional representation:

$$X_{pca} = XV_k \tag{16}$$

where V_k contains the top k eigenvectors and X_{pca} is the projected dataset.

The feature vector was initially 3072 in size, and after applying PCA, we strategically reduced it to 620 features. This reduction aligns with a cumulative explained variance (CEV) of 0.9, indicating that the retained features capture 90% of the variance in the original data. The decision aims to strike a balance between dimensionality reduction for computational efficiency and retaining sufficient information to ensure robust gait pattern recognition in our model. The application of PCA was primarily aimed at reducing the computational complexity associated with a high-dimensional feature space and potentially capturing the most informative components. While PCA does not have an inherent regularization effect like some other techniques, the dimensionality reduction aspect indirectly contributes to mitigating overfitting by focusing on the most salient features.

Classifiers

Our proposed methodology was tested on two different machine learning classifiers: OaA-SVM and random forest. A thorough description of these classifiers is given below.

OaA-SVM

OaA-SVM, which stands for one against all support vector machine, is a well-known multiclass classification technique that expands the capabilities of the conventional support vector machine (SVM) algorithm to include the management of more than two distinct groups[31]. OaA-SVM teaches numerous binary classifiers, each of which is intended to differentiate between one class and the other classes. These binary classifiers are all trained together. During the testing phase, the result of each binary classifier is compared, and the category that had the maximum score is chosen to be the one that will be predicted. In Fig. 7, the architecture of OaA-SVM used in our study is given.

OaA-SVM may be trained through a variety of kernels, which are used to transform the data into a higher-dimensional space in which it can be divided more easily. Some of the most often used kernels are listed below:

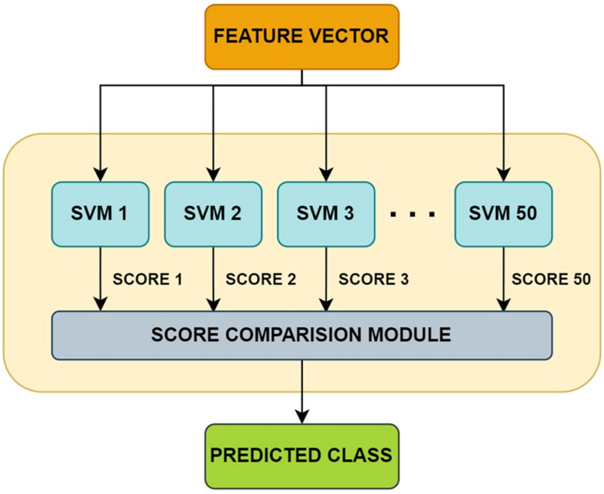

Figure 7. Architecture of OaA-SVM.

1. Linear kernel: This kernel is the most basic kernel used in SVM; it transforms the data linearly. When the data can be separated linearly, it is appropriate.

$$K\left(x_i, x_j\right) = x_i^T x_j \tag{17}$$

2. Polynomial kernel: By transforming the original features of the data using a polynomial function, this kernel performs a nonlinear transformation. This approach is useful when the data have nonlinear boundaries.

$$K\left(x_i, x_j\right) = \left(1 + x_i^T x_j\right)^d \tag{18}$$

Here, d denotes the degree of the polynomial.

3. Radial basis function (RBF): This kernel transforms the information into a higher-dimensional place by applying a Gaussian function to the Euclidean distance between the data points. It is useful when the data are not linearly separable and have complex boundaries.

$$K\left(x_i, x_j\right) = e^{-\gamma |x_i - x_j|^2} \tag{19}$$

Here, γ is a hyperparameter that controls the width of the Gaussian kernel.

The One against All Support Vector Machine (OaA-SVM) was used in our investigation with various kernels. A value of 2 was selected for ’C’ in the linear kernel. The ’C’ value for the RBF kernel was set to 10, while that for the poly kernel was set to 5. These decisions were made with the goal of optimizing the model’s gait recognition performance, taking into account variables such as batch size (32), image size (224*224), and dimensionality (620) following PCA. To balance model complexity and accuracy, we adjusted the parameters. Section "Results and discussion" of this document provides a detailed analysis and results of the experiments conducted with this classifier.

Random Forest

The random forest technique of machine learning is extensively used, and it may be used for classification and regression work[19]. RF is a method that utilizes several decision trees to make accurate predictions. Each tree is trained with different samples of training data and different subsets of input attributes. This helps the system predict new data accurately. When making the final prediction, the algorithm combines all the estimations from each tree. This method reduces the chance of overfitting and improves the model’s accuracy. RF is excellent because it can supervise high-dimensional data and nonlinear relationships between features. It also deals well with outliers and missing data, making it a versatile tool for various situations.

To improve RF performance, one can adjust several hyperparameters, such as the number of trees, the greatest depth of each tree, and the number of attributes considered at each split. Increasing the maximum depth and the number of variables at each split are other ways to boost its effectiveness. Due to its speed and scalability, the random forest algorithm is highly recommended for processing large datasets.

Results and discussion

In this section, the proposed methodology is tested on the CASIA-B-Gait database using different machine learning classifiers. A thorough explanation of the hyperparameters used along with the platform on which the method was evaluated is mentioned.

Implementation details

The dataset used for evaluation is the CASIA-B gait dataset[16]. The dataset is publicly available for use. Due to the resources, the experiment was carried out on 50 participants chosen at random from the dataset; the dataset was split 50–50 for training and testing[32–34]; a thorough description of the dataset is provided in Section "Methodology". The classifiers used for classification were OaA-SVM and random forest.

Experimental Setup: There are several hyperparameters used while making a complete framework; these are given in Table 6 below.

The experiment was carried out on a Google Colaboratory notebook, which had an Intel Xeon CPU, 12.7 GB of RAM, and 15 GB of GPU memory.

Experiment using a Random Forest Classifier

According to Table 7 and Fig. 8, the random forest algorithm achieved accuracies ranging from 97.01% to 90.12% across the 11 distinct angles. The optimal level of accuracy was attained when the angle was set to 0 degrees (97.01%), whereas the minimum level of accuracy was observed at an angle of 126 degrees (90.2%). The accuracy of the random forest model varies from 97.15% to 90.55%. A value of 97.15% was obtained when the angle was adjusted to 0 degrees, which was the ideal level of accuracy. On the other hand, the lowest degree of accuracy was measured at a 126-degree angle and was 90.55%. The random forest method achieved a recall rate ranging from 97.01 to 90.12% according to the recall metric analysis. With a recorded value of 97.01%, the recall performance was determined to be maximal at a 0-degree angle. On the other hand, a 126-degree angle had the lowest recall performance, which was 90.12%. The RF model's F1 score varied between 96.96% and 89.92%. At an angle of 0 degrees, the F1 score attained its greatest value of 96.96%, while at an angle of 126 degrees, it reached its lowest value of 89.92%. According to the results, the random forest approach has promise for recognizing human strides, peculiarly at smaller angles. That this insight is substantial is notable. However, the decreases in the recall, accuracy, precision, and F1 score that were shown at larger angles suggest that this technique has limitations that need to be considered when it is applied in real-world scenarios.

Experiment using the OaA-SVM classifier

For each of the 11 gait angles, Table 8 shows the performance of each classifier in terms of all four assessment metrics: accuracy, precision, recall, and F1 score. With a maximum average accuracy of 98.77%, the linear classifier was demonstrated to be the most effective at distinguishing human gait. This finding indicates that it has the smallest possible margin of error. High accuracy, recall, and F1 score values indicate low rates of false positives and false negatives. As a result, the linear classifier performs well regardless of the gait angle. With a mean accuracy of 96.76%, the polyclassifier executes less well than the linear classifier. This finding suggests that its

Hyperparameter	Value
Image size	224*224
Batch size	32
Input feature vector size	620
OaA-SVM with Linear Kernal 'C' value	2
OaA-SVM with Poly Kernal 'C' value	5
OaA-SVM with RBF Kernal 'C' value	10
No. of trees in Random Forest	500

Table 6. Hyperparameters used in the framework.

Angle	Accuracy	Precision	Recall	F1 Score
000	97.01	97.15	97.01	96.96
018	94.97	95.11	94.97	94.93
036	93.63	94.00	93.63	93.53
054	90.58	90.79	90.58	90.36
072	92.27	92.55	92.27	92.08
090	91.53	91.86	91.53	91.33
108	91.56	91.81	91.56	91.42
126	90.12	90.55	90.12	89.92
144	92.91	93.26	92.91	92.82
162	93.33	93.65	93.33	93.30
180	95.34	95.53	95.34	95.31
Average	93.02	93.30	93.02	92.91

Table 7. Results using a random forest classifier across different angles.

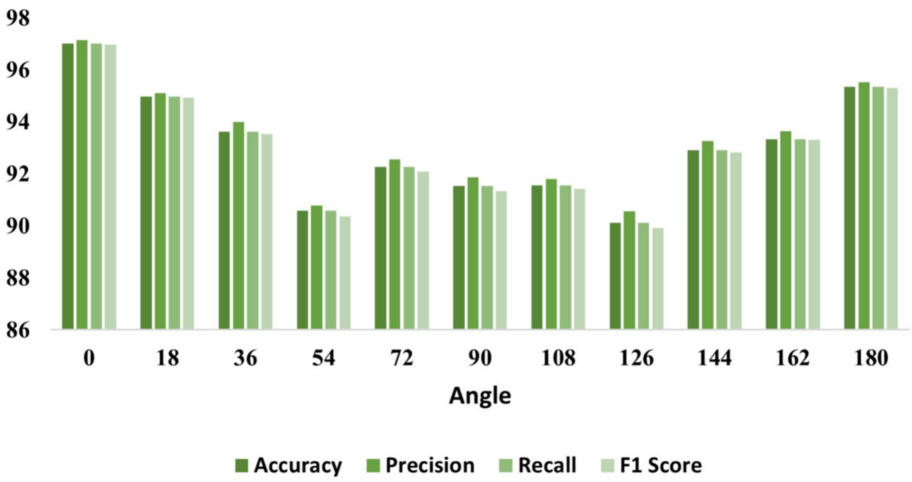

Figure 8. Different evaluation metric results for different angles.

Kernel	Metric	Angles											Mean
		0	18	36	54	72	90	108	126	144	162	180	
Linear	Accuracy	99.10	99.35	98.97	98.23	99.01	99.03	99.12	98.02	98.18	98.98	98.52	98.77
	Precision	99.15	99.36	98.97	98.24	99.03	99.05	99.12	98.11	98.02	98.99	98.56	98.78
	Recall	99.10	99.35	98.97	98.23	99.01	99.03	99.12	98.02	98.18	98.98	98.52	98.77
	F1 Score	99.10	99.35	98.97	98.22	99.01	99.03	99.12	98.01	98.16	98.98	98.50	98.77
Poly	Accuracy	97.85	98.94	98.70	97.76	98.06	98.45	98.33	96.76	96.82	97.76	95.88	97.76
	Precision	98.02	98.94	98.71	97.79	98.10	98.48	98.38	96.86	96.95	97.79	96.04	97.82
	Recall	97.85	98.94	98.70	97.76	98.06	98.45	98.33	96.76	96.82	97.76	95.88	97.76
	F1 Score	97.84	98.94	98.70	97.77	98.05	98.44	98.33	96.73	96.78	97.74	95.82	97.74
RBF	Accuracy	98.44	99.18	98.82	98.20	98.51	98.91	98.86	97.64	97.15	98.46	97.66	98.35
	Precision	98.55	99.19	98.83	98.22	98.54	98.93	98.88	97.77	97.29	98.48	97.71	98.40
	Recall	98.44	99.18	98.82	98.20	98.51	98.91	98.86	97.64	97.15	98.46	97.66	98.35
	F1 Score	98.44	99.18	98.82	98.20	98.50	98.91	98.86	97.61	97.12	98.45	97.63	98.34

Table 8. Results using the OaA-SVM classifier for different angles.

ability to discern human gait patterns is less accurate. It also demonstrates focused values for recall, F1 score, and accuracy, intimating a greater than average proportion of false positives and false negatives. These findings indicate that the poly classifier does not function as well as the linear classifier and has issues with specific gait angles.

Finally, the RBF classifier has an average accuracy of 98.35%. It is more accurate than the poly classifier but still less accurate than the linear classifier at recognizing human gaits. Additionally, the RBF classifier has lower precision, recall, and F1 score values than the linear classifier, although they are higher than those of the poly classifier. These results suggest that the RBF classifier performs better than the poly classifier but not the linear classifier in recognizing human gait.

All the angles of OaA-SVM with a linear kernel performed exceptionally well across various viewing angles, but objectively, a 90-degree angle is one of the most relevant in gait recognition, as the model captures the most relevant features through this angle. A confusion matrix of a 90-degree angle is given below in Fig. 9. In the confusion matrix for the 90-degree angle, each row and column correspond to individual subjects, totaling 50 individuals. This matrix specifically evaluates the model's performance at this particular angle. The false acceptance rate (FAR) is calculated to be 0.00358, indicating a low rate of incorrectly accepting impostors. The false rejection rate (FRR) is recorded as 0.0, signifying that no instances of genuine individuals are falsely rejected. The GAR attains a perfect score of 1.0, demonstrating the model's accuracy in accepting genuine subjects. The Genuine Rejection Rate (GRR) stands at 0.99642, illustrating a high rate of correctly rejecting nonmatching individuals. These metrics collectively highlight the robust performance of the gait recognition model at a 90-degree angle.

Discussion and comparison

The findings demonstrate that in terms of accuracy, precision, recall, and F1 score, the linear classifier outperforms the poly and rbf classifiers. The linear classifier might work well for the dataset because it assumes a straight-line relationship between the input features and the target variable. However, the poly and rbf classifiers use nonlinear decision limits, which can cause overfitting and poor generalizability on the test set. The rbf kernel's sensitivity to its parameter choice also affects performance. These results suggest that the linear classifier

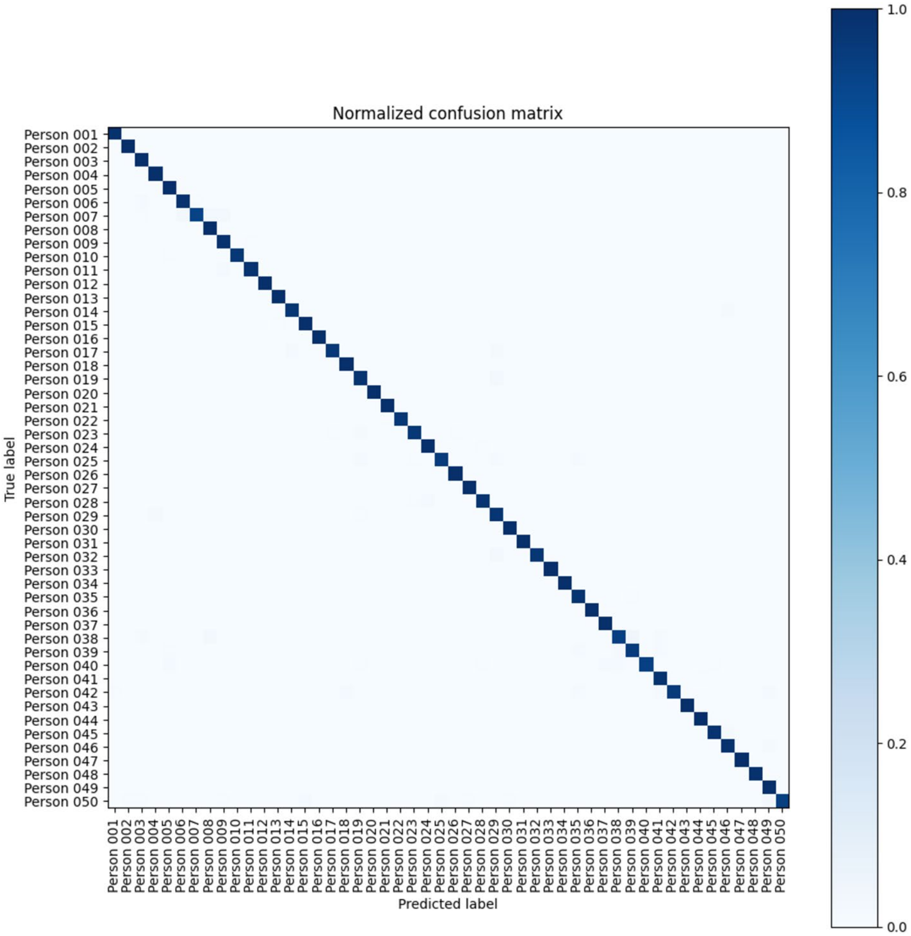

Figure 9. Confusion matrix for a 90-degree angle.

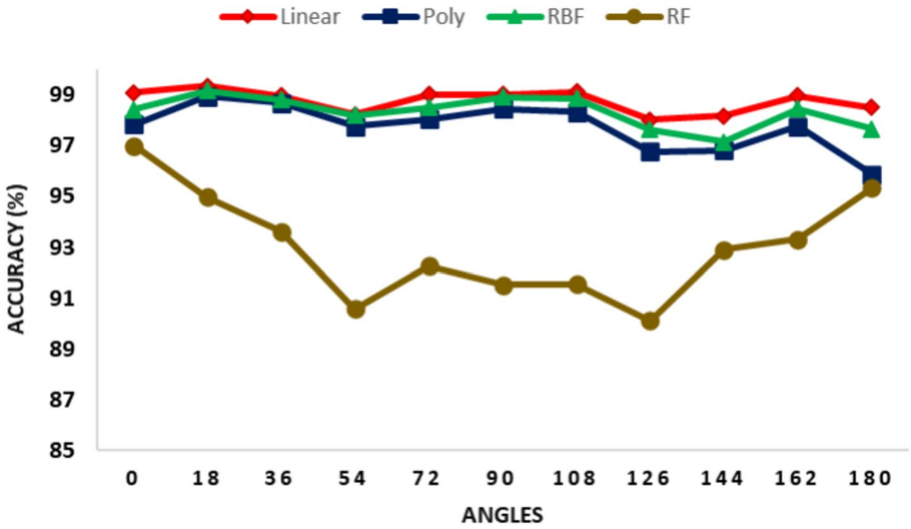

Figure 10. Comparison of different approaches for eleven different bedding angles.

is good for identifying human gait. This is important because gait recognition has many uses, such as biometric identity and surveillance, and a well-performing classifier can make these applications more reliable and effective.

Figure 10 shows that the OaA-SVM classifier performed better in terms of accuracy than did the RF classifier. This might be because the high-dimensional feature space of the gait recognition dataset suits the OaA-SVM classifier better than does the random forest. The linear separation capability of OaA-SVM was deemed more appropriate for the task at hand, as opposed to the decision tree-based approach of the random forest. Moreover, it can be observed that the OaA-SVM classifier exhibits a lower susceptibility to overfitting than does the random forest algorithm. This characteristic may account for its superior performance when evaluated on test data. The line's drop at 54, 126, and 180 degrees—which denotes the models' low performance at these angles—is the significant observation of particular angles. In these domains, more research may improve the models' effectiveness. The variation in walking patterns at particular angles could provide a reasonable explanation for the decreased precision observed at those angles. The subject's body alignment may become more oblique at larger angles, which would enhance the variety of their walking patterns. Variability could make it more difficult for classifiers to distinguish between different people. The quality of the acquired gait data is one such factor that might affect accuracy at various angles. The accuracy of gait data may be questioned under some circumstances, such as low light or occlusion, which will lower the quality of the picture or video. However, the increased accuracy displayed at some angles could be the result of other factors, such as a better camera angle or a gait pattern that is more visible to some people at particular angles. It is probable that certain attributes were more effective than others in helping the classifier identify individuals from different perspectives. Since access control and surveillance systems frequently employ gait detection, our work highlights how crucial security is in this domain. For the purpose of enhancing public safety and security in a variety of circumstances, accurate identification of persons is essential. In addition, we consider the welfare of individuals in many contexts beyond security applications. We hope to make a positive impact on assisted living and healthcare, as well as security measures, by further developing gait recognition algorithms. For instance, accurate identification may have a major impact on medical monitoring and diagnosis, which may enhance patient outcomes and general well-being.

Table 9 presents a comparative analysis of the accuracy of the proposed model against other contemporary models. The results indicate that the proposed model exhibited superior performance in comparison to the other models when tested on a sample of 50 randomly selected data points from the CASIA-B dataset. There are two compelling reasons for the superior performance of our approach in comparison to alternative methods. The approach employed in our study involves the utilization of two highly effective preexisting models in conjunction with PCA for the purpose of reducing dimensionality and the implementation of the OaA-SVM classifier. We conducted experiments with various hyperparameters to determine the optimal configuration, leading to improved accuracy. It is important to understand the limitations of this research, such as the small sample size and the use of just one dataset. Therefore, more research with larger sample sizes and different datasets is needed to confirm the results and prove that the method works well.

Conclusion

The research shows how to use feature extraction and classification for identifying people. The MobileNetV1 and Xception models worked well in obtaining important information from video frames without the need for fine tuning. Combining these features provided a better picture of the data. By reducing the quantity of the data and eliminating superfluous characteristics, PCA improved the efficiency and accuracy of the categorization. High classification accuracy was demonstrated by the OaA-SVM classifier, which had a mean accuracy of 98.77% over three versions and eleven angles. This is important because it demonstrates the use of machine learning for the identification of individuals in security systems such as surveillance cameras. This may result in more sophisticated and practical security systems as well as other applications that require precise human identification. However, this study has several drawbacks. Only 50 participants from the CASIA-B dataset were used in this analysis. Despite being widely used in person identification studies, this dataset may not accurately reflect the overall population and may not be generalizable to other datasets. Therefore, it would be helpful to replicate this work using larger and more diversified datasets to determine whether the recommended technique is generalizable. Furthermore, even though the OaA-SVM classifier showed excellent accuracy, it has certain drawbacks. This classifier, like many machine learning methods, is prone to information biases and cannot work well on datasets with noisy or unequally distributed classes. As a result, it is essential to carefully assess the classifier's performance across a variety of datasets and investigate the possibility of combining several classifiers to increase the overall performance.

Method	No. of Angles	Accuracy (%)
[24] 2021	11	89
[23] 2022	3	95.7
[22] 2022	6	95.83
[18] 2022	11	93.3
Proposed	11	98.77

Table 9. Comparison of state-of-the-art methods with the proposed method.

Data availability

The CASIA-B dataset used for the research is publicly available at http://www.cbsr.ia.ac.cn/.

Received: 12 September 2023; Accepted: 19 July 2024
Published online: 26 July 2024

References

1. Hossain Bari, A. S. M. & Gavrilova, M. L. Artificial neural network based gait recognition using kinect sensor. *IEEE Access* **7**, 162708–162722. https://doi.org/10.1109/ACCESS.2019.2952065 (2019).
2. Potluri, S., Ravuri, S., Diedrich. C. & Schega, L. Deep learning based gait abnormality detection using wearable sensor system. In *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* 3613–3619 (IEEE, 2019).
3. Andersson, V. O. & Araujo, R. M. Person identification using anthropometric and gait data from kinect sensor. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelli*gence 425–431 (2015).
4. Shi, Y. *et al.* Robust gait recognition based on deep CNNs with camera and radar sensor fusion. *IEEE Internet Things J.* https://doi.org/10.1109/JIOT.2023.3242417 (2023).
5. Alharthi, A. S., Yunas, S. U. & Ozanyan, K. B. Deep learning for monitoring of human gait: A review. *IEEE Sens. J.* **19**, 9575–9591. https://doi.org/10.1109/JSEN.2019.2928777 (2019).
6. Katiyar, R., Kumar Pathak, V. & Arya, K. V. A study on existing gait biometrics approaches and challenges. *Int. J. Comput. Sci. Issues* **10**(1), 135–144 (2013).
7. Sepas-Moghaddam, A. & Etemad, A. Deep gait recognition: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**, 264–284. https://doi.org/10.1109/TPAMI.2022.3151865 (2023).
8. Velapure, A. & Talware, R. Performance analysis of fingerprint recognition using machine learning algorithms. *Adv. Intell. Syst. Comput.* **1090**, 227–236. https://doi.org/10.1007/978-981-15-1480-7_19/COVER (2020).
9. De Marsico, M., Petrosino, A. & Ricciardi, S. Iris recognition through machine learning techniques: A survey. *Pattern Recognit. Lett.* **82**, 106–115. https://doi.org/10.1016/J.PATREC.2016.02.001 (2016).
10. Filipi Gonçalves Dos Santos, C. *et al.* Gait recognition based on deep learning: A survey. *ACM Comput. Surv.* **55**, 34. https://doi.org/10.1145/3490235 (2022).
11. Deng, M. & Wang, C. Gait recognition under different clothing conditions via deterministic learning. *IEEE/CAA J. Autom. Sin.* https://doi.org/10.1109/JAS.2018.7511096 (2018).
12. Yeoh, T., Aguirre, H. E. & Tanaka, K. Clothing-invariant gait recognition using convolutional neural network. *2016 International Symposium on Intelligent Signal Processing and Communication Systems, ISPACS 2016*. https://doi.org/10.1109/ISPACS.2016.7824728 (2017).
13. Liao, R., Cao, C., Garcia, E. B., Yu, S. & Huang, Y. *Pose-Based Temporal-Spatial Network (PTSN) for Gait Recognition with Carrying and Clothing Variations.*
14. Yu, S., Tan, D. & Tan, T. A framework for evaluating the effect of view angle, clothing and carrying condition on gait recognition. *Proc. Int. Conf. Pattern Recogn.* **4**, 441–444. https://doi.org/10.1109/ICPR.2006.67 (2006).
15. Jahangir, F. *et al.* A fusion-assisted multi-stream deep learning and ESO-controlled Newton–Raphson-based feature selection approach for human gait recognition. *Sensors* **23**, 2754. https://doi.org/10.3390/S23052754 (2023).
16. Center for Biometrics and Security Research. http://www.cbsr.ia.ac.cn/english/Gait%20Databases.asp. Accessed 12 Apr 2023.
17. Khan, M. A. *et al.* HGRBOL2: Human gait recognition for biometric application using Bayesian optimization and extreme learning machine. *Future Gen. Comput. Syst.* **143**, 337–348. https://doi.org/10.1016/J.FUTURE.2023.02.005 (2023).
18. Arshad, H. *et al.* A multilevel paradigm for deep convolutional neural network features selection with an application to human gait recognition. *Expert Syst.* **39**, e12541. https://doi.org/10.1111/EXSY.12541 (2022).
19. Breiman, L. Random forests. *Mach. Learn.* **45**, 5–32. https://doi.org/10.1023/A:1010933404324/METRICS (2001).
20. Sharif, M. I. *et al.* Deep learning and kurtosis-controlled, entropy-based framework for human gait recognition using video sequences. *Electronics* **11**, 334. https://doi.org/10.3390/ELECTRONICS11030334 (2022).
21. Hofmann, M., Geiger, J., Bachmann, S., Schuller, B. & Rigoll, G. The TUM gait from audio, image and depth (GAID) database: Multimodal recognition of subjects and traits. *J. Vis. Commun. Image Represent.* **25**, 195–206. https://doi.org/10.1016/j.jvcir.2013.02.006 (2014).
22. Khan, M. A. *et al.* Human gait recognition: A deep learning and best feature selection framework. *Comput. Mater. Continua*. https://doi.org/10.32604/cmc.2022.019250.
23. Khan, A. *et al.* Human gait recognition using deep learning and improved ant colony optimization. *Comput. Mater. Continua* **70**, 2113–2130. https://doi.org/10.32604/CMC.2022.018270 (2022).
24. Saleem, F. *et al.* Human gait recognition: A single stream optimal deep learning features fusion. *Sensors* **21**, 7584. https://doi.org/10.3390/S21227584 (2021).
25. Mehmood, A. *et al.* Prosperous Human Gait Recognition: An end-to-end system based on pretrained CNN features selection. *Multimed. Tools Appl.* https://doi.org/10.1007/S11042-020-08928-0/TABLES/7 (2020).
26. Pundir, A. & Sharma, M. A review of deep learning approaches for human gait recognition. In *2023 2nd International Conference for Innovation in Technology (INOCON), Bangalore, India* 1–6. https://doi.org/10.1109/INOCON57975.2023.10101267 (2023).
27. Khan, M. A. *et al.* Human gait analysis: A sequential framework of lightweight deep learning and improved moth-flame optimization algorithm. *Comput. Intell. Neurosci.* https://doi.org/10.1155/2022/8238375 (2022).
28. Howard, A. G. *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. https://doi.org/10.48550/arXiv.1704.04861 (2017).
29. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. *Proceedings—30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017 2017-January* 1800–1807. https://doi.org/10.1109/CVPR.2017.195.
30. Pearson, K. On lines and planes of closest fit to systems of points in space. *Lond. Edinb. Dublin Philos. Mag. J. Sci.* **2**, 559–572. https://doi.org/10.1080/14786440109462720 (1901).
31. Rifkin, R. & Klautau, A. In defense of one-vs-all classification. *J. Mach. Learn. Res.* **5**, 101–141 (2004).
32. Anand, V., Gupta, S., Koundal, D. & Singh, K. Fusion of U-Net and CNN model for segmentation and classification of skin lesion from dermoscopy images. *Expert Syst. Appl.* **213**, 119230. https://doi.org/10.1016/j.eswa.2022.119230 (2023).
33. Anand, V. *et al.* An automated deep learning models for classification of skin disease using Dermoscopy images: A comprehensive study. *Multimed. Tools Appl.* **81**(26), 37379–37401 (2022).
34. Shruti, R. S. & Srivastava, G. Secure hierarchical fog computing-based architecture for industry 5.0 using an attribute-based encryption scheme. *Expert Syst. Appl.* **235**, 121180. https://doi.org/10.1016/j.eswa.2023.121180 (2024).

Acknowledgements

The Chinese Academy of Sciences Institute of Automation's CASIA Gait Database is used in parts of this paper's study. We are grateful to the institution for making this information accessible to the scientific community and advancing the field of gait identification.

Author contributions

In accordance with our authorship policy for Nature (Scientific Reports and Nature Portfolio journals), we provide the following author contributions statement to specify how each author contributed to the manuscript: Akash Pundir: Conceptualization Data curation Methodology Software Writing—original draft Writing—review & editing Manmohan Sharma: Conceptualization Data curation Methodology Software Writing—original draft Writing—review & editing Ankita Pundir: Data curation Validation Writing—original draft Dipen Saini: Methodology Software Formal analysis Salil Bharany: Conceptualization Methodology Supervision Writing—review & editing Khmaies Ouahada: Resources Supervision Writing—review & editing Ateeq Ur Rehman; Resources Conceptualization Supervision Writing—review & editing Habib Hamam : Conceptualization Resources Methodology Writing—review & editing Corresponding Author: Salil Bharany: Conceptualization, Methodology, Supervision, Writing—review & editing.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to S.b. or A.U.R.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.