Artificial Intelligence Review
https://doi.org/10.1007/s10462-022-10365-4

Deep learning pipelines for recognition of gait biometrics with covariates: a comprehensive review

Anubha Parashar[1] · Apoorva Parashar[2] · Weiping Ding[3] · Rajveer S. Shekhawat[1] · Imad Rida[4]

Abstract
This paper presents a comprehensive exposition of deep learning architectures and pipelines for biometric applications using complex characteristics of human gait. The variety and complexity that we have come across encompass the majority of deep learning techniques. Recognizing humans by their walking patterns is a complicated biometric processing approach that identifies people without intervention and works well even with low-resolution images. Several covariates, such as footwear, heavy clothing and carry situations, view angle, occlusion, speed transition, and others, can affect the gait recognition rate. We provide an extensive literature review focusing on the performance of deep learning models in covariate conditions. It shall help researchers to understand specific aspects of deep learning pipelines in gait recognition. The discussion is done on data acquisition, input, dataset, preprocessing, feature extraction, transformation, activation function, classification and training parameters. We also demonstrate the most often used strategies for each parameter over the previous five years. The datasets used so far are compared with the accuracy achieved for each covariate condition. Finally, we listed the benefits and drawbacks of deep learning approaches in covariate conditions along with open problems in the identification based on behavioral traits and concluded the paper by highlighting important lessons.

Keywords Gait recognition · Biometrics · Covariates · Deep learning · Computer vision

Apoorva Parashar, Weiping Ding, Rajveer S. Shekhawat and Imad Rida have contributed equally to this work.

✉ Anubha Parashar
anubhaparashar1025@gmail.com

✉ Weiping Ding
dwp9988@163.com

Extended author information available on the last page of the article

Published online: 18 January 2023

1 Introduction

Biometrics, also known as measures and estimations of unique features of human beings, are the observations and indications of physical attributes of body and behavior such as walk signature or height (Sun et al. 2017). Due to advances in computing, biometrics has become a universal technique to authenticate an individual to ensure secure access to sensitive information and premises and maintain one's privacy. Biometric signatures are categorized as physiological, behavioral, or both, which correspond to the anatomy of the body. Fingerprints, facial features, palm prints, retinal patterns, and odor/scent are a few examples of the former (Jain et al. 2016). Typing speed, keystroke pressure, signature on documents, and gait are behavioral traits that can be uniquely linked to a person (Stylios et al. 2021). The above techniques are widely used to recognize people present in groups and sensitive areas for surveillance purposes.

Gait recognition can be done from a distance without the intervention of the subject and works with low-resolution images, which has picked up the interest of researchers in the field over the last 20 years (Sepas-Moghaddam and Etemad 2022). In earlier days, identity was captured using passwords or cards. However, authentication based on cards and passwords can be easily compromised. Since biometric signatures are exclusive to each individual, they are far more accurate than earlier and ensure that identity is unique and cannot be copied or stolen by anyone. Therefore, biometrics are more secure as reliable evidence of "who you are" and are not prone to duplication. Gait signature is often withstanding to attackers as it is difficult to imitate anyone's gait and is not affected by wearing a mask, cap, gloves, or moving away from the camera (180° from the camera).

The two most common techniques are model-based and model-free. Model-based approaches focus on extracting the gait characteristics by using the anatomy of the subject and joint angles. Such approaches require high computational cost, but they are computing-intensive and necessitate high-resolution pictures (Wolf et al. 2016). Model-free techniques focus on human body motion and often function by extracting gait features from silhouettes directly (Zhang et al. 2017). We have selected all the papers that have developed gait recognition applications. Along with gait recognition, researchers are also using it for reconstruction, adversarial robustness, person identification, age, pedestrian recognition, detection, gender, action recognition, and surveillance.

Deep learning has emerged as a potential approach for effective gait recognition. It might be challenging to start gait recognition using deep learning since one does not know which deep learning pipeline to use or what outcomes to expect. Currently, there are a few review papers that provide details about deep learning algorithms for gait recognition (Connor et al. 2018; Alharthi et al. 2019; Nambiaret et al. 2019). However, some critical factors are missing in these publications, like examining about benefits and limitations of approaches used, reporting the deep learning parameters used in each paper, most adopted approaches in the gait pipeline and comparison of accuracy achieved for a given dataset. The majority of survey papers focused on model-based gait detection systems, putting model-free techniques to the side. Several significant publicly accessible datasets were missing in previous survey papers like NTU RGB+D Dataset, SDUGait database, and more. Factors such as speed variation in walking, spatial and temporal characteristics substantially affect gait recognition findings, yet they are often overlooked.

We tried to encompass most of the factors missing in the previous reviews and included as many deep learning parameters as possible to help researchers that are new to this field. The following are the key contributions of the review paper.

(1) Evaluation of over a hundred gait recognition methods using deep learning with eighteen different parameters for covariate conditions.
(2) Emphasize the most prominent approaches amongst various deep learning pipelines reported.
(3) Present a collection of datasets that were applied in deep learning algorithms in the literature referred.
(4) Highlight deep learning parameters for various gait covariate conditions, which may improve the recognition rate.
(5) Examined the advantages and drawbacks of deep learning based approaches.
(6) List a number of open research questions based on gaps observed in research works so far and propose solutions for a new research direction.
(7) Compare and illustrate the accuracy achieved by a given dataset for various covariate conditions.

The following is a structural breakdown of the paper. Section 1 presents an overview of gait recognition including motivation, contribution, and organization. The structure of a deep learning-based gait identification system is described in Sect. 2. The examination of numerous factors reported in 113 studies led to the development of a deep learning pipeline for gait recognition along with the most adopted techniques of DL pipelines. Section 3 discusses the salient features of DL pipelines along with the benefits and limitations of each approach performed on various covariate conditions. Some open research gaps and their solutions are also proposed for future experiments. Comparative analysis of prominent components of deep learning pipelines is done by looking at the accuracy and dataset used in the research papers. In Sect. 4, we present the conclusion and prospects. In Appendix A in Fig. 6, we provide the acronyms used in this article.

2 Deep dive into deep learning for covariates

Deep learning techniques are very effective at performing various tasks and providing new insights from complex covariate conditions in gait recognition. Figure 1 depicts all possible stages of the deep learning pipeline for gait recognition.

These points are further discussed in the following sub-sections. In most cases, deep learning approaches have the limitation of acting as a black box. For example, if pretrained models are used or hyper tuning is not done, deep learning models rarely provide information about what led them to a particular decision. Table 1 lists deep learning parameters used specifically in covariate conditions for gait recognition. After going through this table, one can quickly determine which deep learning pipeline effectively solves particular covariate conditions.

2.1 Data acquisition and input images

First stage of gait detection involves the collection of gait datasets through various means (sensing technologies). The performance of the deep learning model and its accuracy are heavily reliant on the data quality used. As shown in Fig. 2, gait acquisition can be divided into four broad categories, as per the camera types like day camera (RGB camera, CCTV camera), night camera (depth camera, microsoft kinetic), event-based camera (dynamic vision sensor, event sense camera), wave-based sensors (lidar, radar).

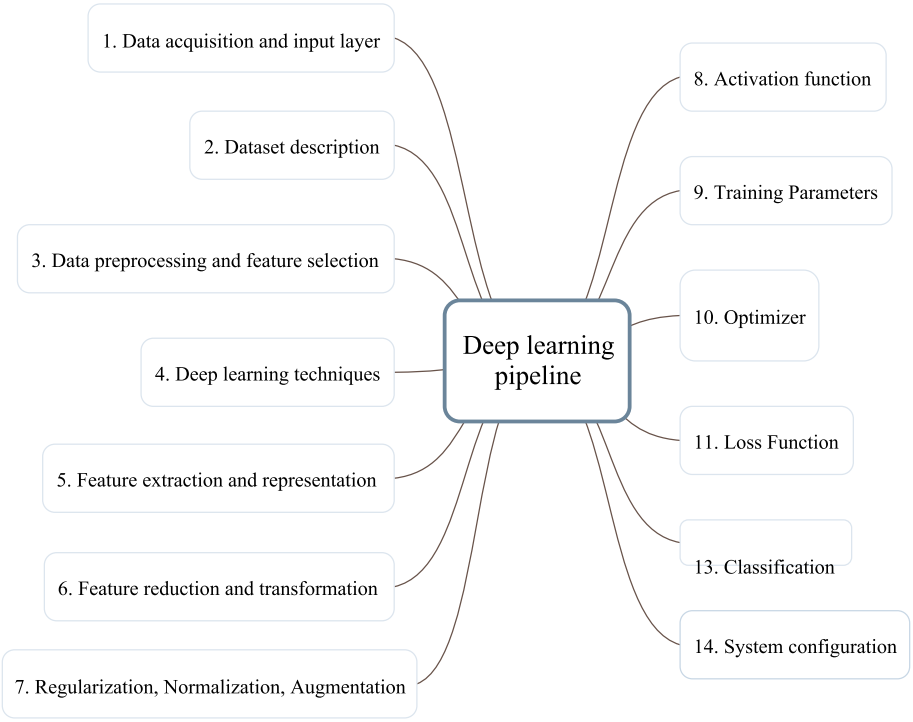


Fig. 1 Deep learning pipeline for gait recognition

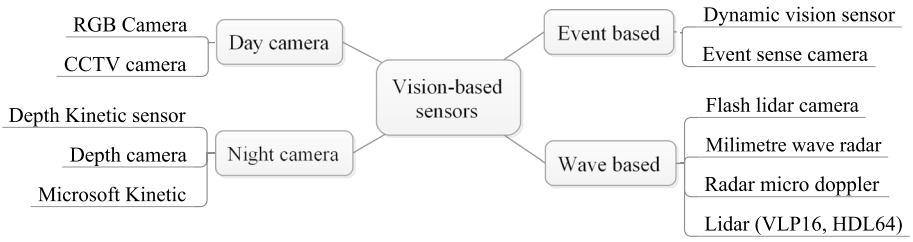


Fig. 2 Categories of sensors used for gait data

Inputs to the gait pipeline can be taken in many forms like images, videos or pre-processed images, and the collection of these images forms a dataset. Cheheb et al. (2018) provided a gallery of samples for training and recognized gait through a probe sample for testing. One of the important input technique is Gait Energy Image (GEI). GEI is a large descriptor that encapsulates many details by averaging a gait cycle (Xu et al. 2019). GEI retains the spatial detail of a gait series and is one of the many ways to get information from cameras (Babaee et al. 2018a). Occlusion is a common occurrence that takes place when two or more persons get too near to one another and seem to integrate or become one. Gait recognition systems often track obscured frames incorrectly.

Table 1 List of deep learning parameters used in gait-based covariate papers

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Babaee et al. (2018a)	GEI	OULP / 86.3	Model Free	RGB camera	CNN model	CNN	Up-sampling	ReLU
Babaee et al. (2018b)	Incomplete GEI	OULP / 86.3	Model Free	RGB camera	FCN	GAN	NA	NA
Babaee et al. (2019a)	Incomplete GEI	OULP (90°)/ 80	Model Free	RGB camera	GAN Generator Encoder Decoder	Generator	Discriminator	LReLU, Tanh sigmoid,ReLu
Xia et al. (2019)	Gait Silhouettes	CASIA-B / 96.20 TUM-GAID/ 95.60	Model Free	RGB camera	GAN Generator Discriminator	BS / Encoder	Sparse representation	Nonlinear
Babaee et al. (2019b)	Incomplete GEI	OULP/ 94. 10; CASIA B / 85.30,	Model Free	RGB camera	Convolution Deconvolution	GEI / Encoder	PCA Unsampling	Sigmoid
Li et al. (2019a)	GEI	OULP-Bag / 93.85	Model Free	RGB camera	GAN	CNN	NA	LeakyReLU
Das et al. (2019)	GEI	TUM-IITKGP/ 98.89 CASIA-B/ 93.39	Model Free	RGB camera	VGG-16, LSTM, GAN	Zero centred Gaussian noise/ Encoder-	PCA	Softmax
He et al. (2019)	Probe Template	OULP / 93.1 CASIA B/ 74.6 USF / 94.7	Model Free	RGB camera	GAN; Generator Discriminator	Mean-pooling	View transform	LeakyReLU
Uddin et al. (2019a)	Frames-low resolution	OU-MVLP / 82.40	Model Free	RGB camera	Generator; Encoder-Decoder	GEI, Encoder, Decoder	NA	Rectified Linear Unit
Wang et al. (2019a)	Gait sequences	CASIA-B / 95	Model Free	RGB camera	CNN	GEI / CNN	Flip, Rotation, Zoom, Clip, Translation, Adding Noise	Rectified Linear Unit

Table 1 (continued)

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Yeoh et al. (2016)	Gait Energy Image	OU-ISIR Treadmill B / 81.60.5	Model Free	RGB camera	CNN-SVM	Spatiotemporal representation/ CNN	NA not mentioned	Rectified Linear Unit
Alotaibi and Mahmood (2017)	Frames	CASIA-B Average - 98.4 NM 86.7 BG 94.8 CL 93.3 ; OULP 53.7	Model Free	RGB camera	Deep learning	CNN	LDA	Sigmoid; ReLU; LReLU
Castro et al. (2017a)	Silhouettes	TUM-GAID/ 89.00	Model Free	RGB camera	CNN - SVM	Spatio-temporal OF / CNN	NA	ReLU
Tong et al. (2017)	Pair of GETs	CASIA B / 86.14	Model Free	RGB camera	CNN	DCNN	NA	NA
Yeoh et al. (2017)	GEI	OUISIR Treadmill B/ 94.8	Model Free	RGB camera	Stacked Progressive Auto-Encoders	Encoder	PCA	Tanh
Sun and Liu (2018)	Image	CASIA B/ 96.1	Model Free	RGB camera	VGG 16, Faster R-CNN	CNN	NA	Softmax
Yang et al. (2019)	Image	CASIA-B/ 99.19	Model Free Model based	RGB camera	CNN - ResNet, LSTM	Coordinates concate- nation / GEI skeleton	NA	Sigmoid
Li et al. (2019b)	GEI	OU-LP-Bag / 89.6; TUM-GAID/ 99.7	Model Free	RGB camera	Deep learning architecture	Encoder-decoder	NA	ReLU
Ling (2019)	GEI	OU-ISIR TEADMILL B/ 89.2	Model Free	RGB camera	CNN	CNN	Discriminative feature selector	Sigmoid
Castro et al. (2019)	Frames	TUM-GAID / 86.00	Model Free	RGB camera	CNN	CNN	NA	ReLU

Table 1 (continued)

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Li et al. (2020a)	GEI	CASIA B/ BG 100; CL 95.6; OU-LP-Bag/ 87.04	Model Free	RGB camera	Encoder-decoder	Euclidean distance / CNN	NA	ReLU
Luo and Tjahjadi (2020)	RGB	CMU MoBo / 95.2; CASIA B / 88, KY4D / 92; OU-MVLP /87; OULP / 94	Model Free Pose	RGB camera	CNN	CNN	NA	Sigmoid
Li et al. (2020b)	Image	OU-LP-Bag/ 91.2; CASIA B/ 98.3	Model Free	RGB camera	Generator, Discriminator, CNN, GAN	Encoder	Decoder	ReLU; LReLU Sigmoid
Yan et al. (2015)	Silhouettes	CASIA B / 95.88	Model Free	CCTV camera	CNN	GEI / Temporal Spatial changes	NA	ReLU
Yu et al.(2017)	Silhouettes	CASIA-B/ 62.8	Model Free	RGB camera	GaitGAN EncoderDecoder	GEI/Encoder	Decoder;	LReLU
Yu et al. (2019)	Silhouettes	CASIA-B/ 96.30	Model Free	RGB camera	Gan Encoder Decoder	Encoder/ reconstructed GEI	Decoder;	LReLU
Alotaibi and Mahmood (2017)	Silhouettes	CASIA-B/ 86.7	Model Free	RGB camera	Deep CNN architecture	CNN	NA	Sigmoid ReLU, LReLU
Yu et al. (2017)	Gait energy image	CASIA B/ 97.58 nm. 72.14 bg; 45.45 cl. Average - 63.90%	Model Free	RGB camera	Deep model auto-encode	Invariant feature extraction	PCA	Sigmoid
Liao et al. (2017)	Frames	CASIA B / 96.92 nm 85.78 bg ; 68.11 cl	Model free Pose based	RGB camera	CNN-LSTM; pose estimation	CoordinateExtraction / CNN - LSTM	NA	Hidden activation

Table 1 (continued)

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Yao et al. (2018)	Frames	CASIA B / 75	Model Based	RGB camera	Model based plus CNN	Skeleton extraction/CNN	Gaussian distribution	ReLU; Tanh
Cheheb et al. (2018)	Video sequences	CASIA B/ 98.81 nm; 75.72 bg; 65.44 cl	Model Free	RGB camera	Encoder Decoder Deep network	Silhouette extraction/ SSA	NA	Softmax
Zhang et al. (2019a)	RGB silhouettes	FVG/ 87.8; CASIA B/ 91.4; USF 99.5	Model Based	Flash lidar camera	LSTM Encoder Decoder; CNN	GEI / Encoder Decoder; Sequence pose features	NA	ReLU, Sigmoid
Thapar et al. (2019)	Binary silhouettes	CASIA-B / 99.16 nm 89.5 bg 90.2 cl; OULP / 99.00	Model Free	RGB camera	LSTM-VGRNet2 pretrained 3D CNN	SSM / 3-D CNN	NA	Softmax
Hawas et al. (2019a)	Silhouettes	CASIA-B/ 95	Model Free	RGB camera	CNN	GEI / CNN	Optical flow computation	ReLU
Hawas et al. (2019b)	GEI Skeleton	CASIA-B/ 92.3 nm; 88.9 bg; Cl 62.3 FVG / 96.3	Model free Model based	RGB camera	Encoder Decoder, LSTM	CFA; PFA / Encoder	Decoder structure	Leaky ReLU
Castro et al. (2020)	OF depth gray pixels	TUM-GAID/ 96.5 CASIA-B / 87	Model Free	RGB camera	ResNet	Gait descriptors	Pooling	ReLU, Softmax
Mehmood et al. (2020)	RGB frame	CASIA B / 94	Model Free	RGB camera	Pre-trained CNN Densent-V3	Remove noise; resize data/ CNN	YCbCr Transformation	ReLu
Swee et al. (2014)	Silhouette	CASIA-B / 79 OULP / 91	Model Free	RGB camera	GAN - CNN	GEI Reconstruction through GAN/ CNN	Discriminator	ReLU

Table 1 (continued)

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Shiraga et al. (2016)	GEI	OU-MVLP / 55 65 75; 80.4 91.5 94.8	Model Free	RGB camera	CNN	CNN	NA	ReLU
Wolf et al. (2016)	Silhouette	CMU MoBo - 99; USF/ 89; CASIA B - 99	Model Free	RGB camera	CNN	CNN	NA	ReLU
Zhang et al. (2017)	Gait sequences	OULP - 96.2; CASIA B 95.7	Model Free	RGB camera	pretrained VGG-D net	Region of Interest/ CNN	PCA	NA
Li et al. (2017a)	Gait sequences	OULP / 88.7	Model Free	RGB camera	CNN	VGG-D	PCA	NA
Jia et al. (2017)	GEI	CASIA B / 97.3	Model Free	RGB camera	5-layer CNN	CNN	pooling by a rectifier; LDA	ReLU
Thapar et al. (2018)	Video frames	CASIA B / 98.75	Model Free	RGB camera	3D CNN's	CNN	NA	ReLU
Zhang et al. (2019b)	Silhouette	CASIA B / 73	Model Free	RGB camera	GAN	Generator/ encoder Decoder	Discriminator	ReLU
Zhang et al. (2019c)	Silhouette	CASIA- B / 74	Model Free	RGB camera	GAN	Generator/ cross-gait recognition	Discrimination	ReLU
Deng et al. (2020)	Gait sequences	CASIA-B / 87-93	Model Free	RGB camera	DCRNN	CNN	NA	Rectified Linear Units
Wang et al. (2019b)	GEI	CASIA B / 66.2; OULP / 93.1	Model Free	RGB Camera	GAN	Generator/ CNN	Discriminator	NA
Wu et al. (2016) 2017	GEI	CASIA B / 98; OULP / 78.3; USF / 96.3	Model Free	RGB camera	DCNNs	BS based on GMM / CNN	NA	ReLUs
Chen et al. (2017)	Silhouette	OULP / 94.9	Model Free	RGB camera	CNN	Fully Convolutional Neural Network	NA	ReLU

Table 1 (continued)

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Huang et al. (2018)	Silhouette sequences	OULP / 89.2; OU-MVLP / 96.2	Model Free	RGB camera	CNN	GEI / CNN	NA	Sigmoid; ReLU
Hu et al. (2048)	GEI	OU-MVLP / 79; CASIA-B / 93.2	Model Free	RGB camera	Discriminant Gait; CNN	Angle Sensitive Discriminator/ CNN	NA	NA
Luo et al. (2019)	Gait Sequence	CASIA-B/ 96.10	Model Free	RGB camera	CNN	CNN	NA	NA
Linda et al. (2020)	Gait sequences	CASIA-B/ 94.65	Model Free	RGB camera	CNN	BS / Color-mapped Contour	Linear interpolation	ReLU
Ben et al. (2019)	GEI	CASIA B / 98.8 OULP / 93.78	Model Free	RGB camera	Coupled bilinear	discriminant projection	NA	NA
Tong et al. (2019)	GEI	CASIA B / 93.78; OULP 94.62; USF/ 93.87	Model Free	RGB camera	CNN	CNN	NA	ReLU
Zhang et al. (2019d)	Silhouette	CASIA-B / 88, OULP / 99 OU-MVLP/ 67.1	Model Free	RGB camera	LSTM	Spatial-temporal feature learning CNN	NA	Softmax
Wang et al. (2020a)	Gait silhouette	OULP / 93	Model Free	RGB camera	CNN	Object region segmentation/ CNN	Max pooling	ReLU
Chao et al. (2019)	Video sequence	CASIA- B / 95.0 nm, 87.2 bag, 70.4 coat; OU-MVLP 87.1	Model Free	RGB camera	CNN -Multilayer Global Pipeline	Multi-level sequence features	Set pooling	ReLU; Softmax

Table 1 (continued)

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Zhang et al. (2019d)	Multiple pair of gaits	OULP-Bag/ 95.8 CASIA-B / 98.7	Model Free	RGB camera	JUCNet	CNN pooling	Ablation	Softmax
Takemura et al. (2017)	GEI	OU-MVLP/ 98.8	Model Free	RGB camera	CNN	CNN	NA	ReLU
Zhang et al. (2019e)	GEI	OULP /100; CASIA B/ 70; OU-MVLP 96.2	Model Free	RGB camera	GAN-CNN-VGG16	GAN generator/ CNN	Discriminator	LeakyReLu
Zhang et al. (2019f)	GEI	CASIA-B/ 75.0; OULP / 97.9	Model Free	RGB camera	CNN	CNN	NA	NA
Wang and Yan (2020)	Silhouette	CASIA B / 95.9; OULP/ 99.1	Model Free	RGB camera	CNN-LSTM	frame-by-frame GEI/ CNN	NA	ReLU
Khan et al. (2020)	Frames	CASIA-B / 98.5 OULP / 98.9	Model Free	RGB camera	Nonlinear Deep Neural Network	Cross-view gait descriptor / NDNN	PCA	LeakyReLU
Xu et al. (2020)	GEI	OU-MVLP/ 63.1; CASIA B/ 92.7; OULP/ 98:93	Model Free	RGB camera	CNN (GAN, discriminative)	CNN	FFD geometric transformation	ReLU
Cai et al. (2019)	Frame	CASIA B/ 96.181	Model Free	RGB camera	CNN	Multi-level sequence features	Set pooling	ReLU; Softmax
Feng et al. (2016)	Silhouette	Human 3.6M / 81.9	Model Free Pose based	RGB camera	LSTM - CNN	Heatmaps / LSTM	NA	ReLU
Haque et al. (2016)	4D input data	Depth Silhouettes / 90	Model Free	Depth camera	LSTM - CNN	Encoder layer	Linear transformation	Softmax

Table 1 (continued)

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Uddin et al. (2019b)	Depth Video	Depth Silhouettes/ 98.5	Model Free	Depth camera	CNN	Silhouette extraction / Optical Flows	Local Directional Pattern	ReLU
Li et al. (2017b)	Frames	NTU RGB+D / 82.1 NTU RGB+D 120 / 75.2	Model Based	RGB camera	DCNN (VGG-19)	mean-subtracting/ Savitzky-Golay smoothing filter	NA	ReLU
Castro (2017b)	RGB video frames	TUM GAID/ 94.3	Model Free	RGB camera	CNN - ResNet	Optical Flows / CNN	NA	ReLU; PReLU
Costilla-Reyes et al. (2018)	Footstep signals	Floor Sensor System/ 93	Sensor Based	Piezoelectric sensors	Deep Residual Network Model	Spatial Temporal / CNN	NA	ReLU
Karianakis et al. (2017)	Silhouette	RGB-D / 78.6; Depth ReID/ 91.1 skeleton data TUM GAID / 97; FaceBody / 92.9	Model Free	Depth Kinect sensor	DCNN Encoder	Cartesian distance / encoder layer	NA	Sigmoid
Batchuluun et al. (2018)	Image	CASIA C/ 99.71	Model Free	RGB camera	CNN-LSTM; VGG	BS; binarization/ Detection, COG	Euclidian distance	ReLU
Liu et al. (2018)	GEI	OULP / 89.54	Model Free	RGB camera	3D-CNN; 2D-CNN; ResNet-18	CNN	NA	Fully connected
Tong et al. (2018)	GEI	CASIA-B/ 93.64, OULP / 95.67 CMU MoBo / 92.71	Model Free	RGB camera	CNN - LSTM	CNN	NA	Tanh

Table 1 (continued)

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Karianakis (2018)	RGB and depth data	TUM GAID/ 76.3; BIWI / 50.0; IIT PAVIS/ 52.4;	Model Free	Depth Kinect sensor	DCNN	Background subtraction/ CNN	NA	Bernoulli-sigmoid; Relu
Battistone and Petrosino (2019)	Video frames	3D/ 95.2; CAD-60/ 94.4; CASIA B/ 86.4; TUM-GAID/ 98.4	Model free Model based	RGB camera	Time based Graph - LSTM	Gaussian low-pass filter with 0 mean	Pooling	Sigmoid; Softmax; ReLU
Wu et al. (2020)	Silhouette	OU-ISIR Treadmill B /67; CASIA B/ nm - 92.1; bg- 76.8; CL - 58.2	Model Free	RGB camera	CNN	CNN	NA	LeakyReLU
Huynh-The et al. (2020)	2D/3D posture	UPCV Gait/ 98.86; UPCV Gait K2 / 99.65; S20/87.63; SDUGait/ 91.04	Model Based	RGB camera	CNN	Segmentation, alignment/ CNN	NA	ReLU
Wu et al. (2015)	GEI	CASIA-B/ 84.85	Model Free	RGB camera	A pre-trained DenseNet	CNN	NA	ReLU
Fan et al. (2020)	Silhouette	CASIA-B/ 78.7; OU-MVLP 88.7	Model Free	RGB camera	FCL	Frame-level Part Extractor/ CNN	Horizontal Pooling	Leaky ReLU
Liu and Liu (2020)	GEI	CASIA-B /97.58; UCMP-GAIT/ 92.22	Model Free	RGB camera	CNN	Encoder	Decoder	ReLU

Table 1 (continued)

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Rauf et al. (2016)	Silhouette	CASIA B/ 96	Model Free	RGB camera	CNN	CNN	NA	Sigmoid; Tanh; ReLU
McLaughlin et al. (2016)	Video frames	iLIDS-VID/ 95; PRID-2011/ 96	Model Free	RGB camera	CNN - RNN	Optical flow/ CNN	NA	Tanh
Sokolova and Konushin (2017)	OF frames	CASIA B/ 74.93; TUM-GAID/ 97.20	Model Free	RGB camera	CNN - VGG	Descriptor	PCA transformation	ReLU
Marín-Jiménez et al. (2017)	OF frames	TUM-GAID/ 63.6	Model Free	RGB camera	CNN	CNN	PCA	Rectified Linear Unit
Nithyakani et al. (2019)	GEI	TUM- GAID/ 90.6, 94.7	Model Free	Depth camera	CNN; DCNN	CNN	NA	ReLU
Carley et al. (2019)	Silhouette	OULP/ 58	Model Free	RGB camera	CNN	CNN	NA	NA
Song et al. (2019)	RGB frames	CASIA B/ 89.9; SZU RGB-D/ 78.5; Outdoor-Gait / 85.2	Model Free	RGB camera	CNN	Silhouette; GEI/ CNN	Deconvolutional layer	Sigmoid
Arshad et al. (2022)	RGB frames	CASIA A/ 99.7; CASIA B/ 93.3; CASIA C/ 92.2; AVAMVG/ 99.8	Model Free	RGB camera	DCNN; AlexNet; VGG19	CNN	LDA	NA
Delgado-Escaño et al. (2020)	augmented version	Generates two augmented datasets / 99.7, 98.7	Model Free	RGB camera	CNN	Crop & Stack OF, segmentation/ OF	Aggregation	NA
Jia et al. (2019)	Silhouette	CASIA-B/ 82	Model Free	RGB camera	GAN - Residual Block	CNN	Deconvolution Layer	ReLU

Table 1 (continued)

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Wang et al. (2020b)	Frames	ESIM-RW/ 99; Ev-RW/ 78	Model Free	Event sense camera	CNN - SOTA SR networks	VGG 19; ResNet	Three discriminators	ReLU
He et al. (2020)	Silhouette	CASIA B/ 76.8	Model Free	RGB camera	GAN	Adversarial frames	NA	ReLU
Xue et al. (2020)	Silhouette	CASIA B/ 65.1; OULP/ 89.1	Model Free	RGB camera	GAN - CNN	CNN-generator / CNN- Discriminator	NA	LeakyReLU
Xu et al. (2019)	GEI, CGI	CASIA-B/ 74.44; OU-ISIR Treadmill dataset B/ 85	Model Free	RGB camera	Capsule network ; CNN	Spatio-temporal features, i.e., CGI	NA	ReLU
Zheng et al. (2016)	Frames	PRID-2011/ 74.8; iLIDS-VID/ 51.3; MARS/ 65.3	Model Free	RGB camera	Deformable CNN	CNN – HOG 3D / GEI	NA	ReLU
Makihara et al. (2018)	GEI	OULP / 91	Model Free	RGB camera	Deformable CNN	chroma-key/ aggreg-ated gait feature	NA	ReLU
Sakai et al. (2019)	Gait silhouette	OULP / 94.10	Model Free	RGB camera	CNN	CNN	PCA	ReLU
Zhu et al. (2018)	Input Image	Caltech/ 93.06; INRIA/ 94.34 ETH/ 75.14	Model Free	RGB camera	DNN	CNN	NA	NA

Table 1 (continued)

Cite/year	Input	Dataset/accuracy in %	Types	Data acquisition	Gait pipeline	Pre-processing/ feature extraction	Transformation	Activation function
Alharthi et al. (2019)	Raw video	CASIA B/ 48.57% to 86.25% depending on the view angle	Model Free	RGB camera	GAN	Pre-processed original gait and the noise gait to extract their silhouette contours / GEI - CNN	NA	ReLU

Cite/year	Classification	Regularization augmentation	Optimizer	Training parameters	Loss function	GPU/CPU	Covariates handled
Babaee et al. (2018a)	Sigmoid	BN; Dropout - 0.5	Adam	SGD	Mean square error	GPU	Occlusion
Babaee et al. (2018b)	NA	NA	NA	NA	NA	CPU	Occlusion
Babaee et al. (2019a)	Binary classification	Pixel values are normalized	Adam	LR = $10^{-3}, 10^{-4}$	Mean square error	GPU	Occlusion
Xia et al. (2019)	Back-propagation	NA	RMSprop	SGD LR = 0.00005, 0.01	Mean square error	GPU	Occlusion
Babaee et al. (2019b)	Nearest neighbour	Normalization, Dropout - 0:5	Adam	SGD LR = 0.8, 10 X 10^{8}	Mean square error	CPU	Occlusion
Li et al. (2019a)	Regression based		Adam	LR = 0.0001,	Adversarial; Pixel-wise	CPU	Carrying
Das et al. (2019)	Random Forest	Dropout - 0.2, BN	RMSprop optimizer	Skip connection; LR= 0.001	Binary-cross entropy	GPU	Occlusion
He et al. (2019)	Softmax; NN	L2 norm	Discriminator	LR= 0.01, 5×10^{-5}	Adversarial; Cross-entropy	CPU	View
Uddin et al. (2019a)	Wasserstein GAN	Batch normalization	Adam	LR= 0.0001	Triplet hinge; Adversarial; Reconstruction	CPU	Occlusion

Table 1 (continued)

Cite/year	Classification	Regularization augmentation	Optimizer	Training parameters	Loss function	GPU/CPU	Covariates handled
Wang et al. (2019a)	Softmax	Dropout of 0.5	NA	NA	NA	CPU	Occlusion Pedestrian View
Yeoh et al. (2016)	Linear SVM	LRN, dropout	NA	SGD M= 0.8. LR= 0.01. Mean = 0	Euclidean distance	GPU	Clothes
Alotaibi and Mahmood (2017)	NC; Softmax	Duplicate; Rotate;	RMSP	LR= 0.001	NA	CPU	Clothes View Carrying
Castro et al. (2017a)	Softmax , SVM	L2-normalize; Dropout= 0.4	NA	SGD M= .9,.95 LR= $10^{\wedge 2}$ WD= $5x10^{-4}$	NA	GPU	Clothes
Tong et al. (2017)	Logistic regression loss	NA	Probability distribution	SGD	Contrastive loss.	CPU	Clothes
Yeoh et al. (2017)	Sigmoid	Linear transformation		SGD	Least square error	CPU	View Clothes Carrying
Sun and Liu (2018)	SVM, Softmax	NA	Adam	LR= 0.01	Cross-entropy loss	GPU	Clothes
Yang et al. (2019)	Softmax at 0.75		Adam	LR= 0.0001	Binary cross-entropy	GPU	Clothes; Carrying
Li et al. (2019b)	SVM	L2 norm; LRN; Dropout= 0.5	Linear optimization	SGD M=0.9 LR= 0.01, four times divide by 10	Contrastive, Triplet loss	GPU	Clothes; Carrying Spatial
Ling (2019)	Latent semantic analysis	NA	Euclidean distance	NA	Cross-entropy loss	CPU	Clothes
Castro et al. (2019)	Softmax	Dropout	Binary Connect	SGD	Back propagation	GPU	Challenging computation
Li et al. (2020a)	Cross-reconstruction	Translation	Adam	LR= 0.0002 M = 0.5, 0.999	Cross-entropy loss	CPU	Carrying; View Clothes; Speed
Luo and Tjahjadi (2020)	Softmax	NA	Residual error cost function		Cross-entropy	GPU	Camera; View Clothes

Table 1 (continued)

Cite/year	Classification	Regularization augmentation	Optimizer	Training parameters	Loss function	GPU/CPU	Covariates handled
Li et al. (2020b)	Discriminator	L1 norm; Dropout= 0.5	Adam	LR= 0.0002 M= 0.5	Contrastive, Triplet loss	CPU	Carrying
Yan et al. (2015)	Back-propagation	Dropout; Data augmentation	Regularizing	SGD; LR= 0.01 retraining task-specific M= 0.6 WD = 0.0005	Joint loss function	GPU	Clothes View
Yu et al.(2017)	Discriminative model	NA	NA	NA	Binary cross entropy	CPU	View; Clothes Carrying
Yu et al. (2019)	Real/fake discriminator	NA	Inc interclass dist.; dec intra -class dist.	NA	Contrastive loss	CPU	View; Clothes Carrying; Posture
Alotaibi and Mahmood (2017)	Softmax	NA	RMSP	SGD with Adaptive LR	Cross entropy error	CPU	Occlusion Clothes; View
Yu et al. (2017)	Nearest neighbor	Auto-encoder for image transformation	Least square error	SGD LR= 0.1	Euclidean loss	CPU	View; Clothes Carrying
Liao et al. (2017)	NA	NA	Multi-loss strategy	NA	Cross-entropy, Contrastive loss	CPU	Clothes; Carrying
Yao et al. (2018)	Softmax	Local response normalization	Momentum optimization	BP; LR=.01 multiply .95 for10k epoch	CCE, Regression loss	CPU	View; Clothes
Cheheb et al. (2018)	Softmax	NA	NA	NA	NA	CPU	View; Carrying
Zhang et al. (2019a)	Auto encoder	BN, Similarity loss	Adam	LR= 0.0001	Incremental identity, Cross reconstruction	GPU	Clothes Carrying; View Spatial Temporal
Thapar et al. (2019)	LSTM classifier	NA	Adam	LR= 1 e −5	Triplet loss	CPU	View; Carrying Clothes

Table 1 (continued)

Cite/year	Classification	Regularization augmentation	Optimizer	Training parameters	Loss function	GPU/CPU	Covariates handled
Hawas et al. (2019a)	Softmax	Local response normalization	NA	SGD LR=0.01	NA	CPU	View; Clothes
Hawas et al. (2019b)	Sigmoid	Weights decay= 0.001	Adam	LR= 0:0001 M= 0.9.	Pose similarity, Canonical Consistency	GPU	Clothes; Carrying View
Castro et al. (2020)	Directed Acyclic Graph	NA	NA	NA	NA	GPU	Noise
Mehmood et al. (2020)	Multi SVM; KNN	Batch Normalization	Manhattan Distance	LR= 0.0001	cross entropy loss	GPU	Clothes; View
Swee et al. (2014)	Softmax	Instance Normalization	NA	BP	Adversarial, cross entropy	CPU	View
Shiraga et al. (2016)	Softmax	Local response normalization	NA	SGD	Cross entropy	CPU	View
Wolf et al. (2016)	Softmax	Dropout= 0.5	NA	SGD, LR=10^{-4} /10^{-5} M=.9; WD=5*10^{-4}	NA	CPU	View
Zhang et al. (2017)	Nearest neighbor	L2 norm; NAC	NA	NA	Euclidean distance	CPU	View Clothes; Carrying
Li et al. (2017a)	Nearest neighbor	L2 norm; NAC	2-fold cross validation	NA	Similar score, ED	CPU	View Clothes; Carrying
Jia et al. (2017)	Decision based	Dropout	Feature Optimization	NA	Euclidean distance	CPU	View
Thapar et al. (2018)	Softmax	Dropout= 0.5	NA	SGD LR= 0:005	NA	CPU	View
Zhang et al. (2019b)	Nearest Neighbour	NA	Adam	LR= 0.0002	Contrastive, ED	CPU	View
Zhang et al. (2019c)	Triplet loss	NA	Adam	NA	Adversarial Loss	CPU	View

Table 1 (continued)

Cite/year	Classification	Regularization augmentation	Optimizer	Training parameters	Loss function	GPU/CPU	Covariates handled
Deng et al. (2020)	Softmax	NA	NA	NA	NA	CPU	View
Wang et al. (2019b)	Softmax	NA	NA	LR= 10^{-5}	GAN Loss	CPU	View Cross-view
Wu et al. (2016) 2017	Softmax	NA	Euclidean distance	BP LR= 0.01	Logistic regression loss	CPU	Multi-view Cross-view Crosswalk
Chen et al. (2017)	Softmax	Data augmentation	NA	SGD LR=.001; M= 0	Contrastive loss	GPU	Cross-view
Huang et al. (2018)	Softmax	L2 normalization	Objective function	SGD; M= 0.9; LR= 0.0008	ED, Triplet Contrastive	CPU	Cross-view
Hu et al. (2048)	Triplet loss	NA	Objective function	NA	Reconstruction loss	CPU	Cross-view
Luo et al. (2019)	Triplet is set to 0.2	NA	Adam	LR=$1e^{-4}$	Softmax loss	GPU	Cross-view
Linda et al. (2020)	Softmax	2D Gaussian functions	Adam	LR=$1e^{-08}$	CCE; BCC	CPU	Cross-view
Ben et al. (2019)	Metric learning	NA	Objective function	NA	NA	GPU	Cross-view
Tong et al. (2019)	LR; Softmax	NA	Restrictive triplet loss	NA	Triplet loss	CPU	Cross-view
Zhang et al. (2019d)	softmax	L1norm; COCO, CosFace and ArcFace loss	SGD	LR=0.1	Angle centre loss	GPU	Cross-view
Wang et al. (2020a)	MCNN based classification	Silhouette normalization	NA	NA	BP; Cross entropy loss	CPU	Cross-view
Chao et al. (2019)	Horizontal pyramid pooling	Triplet sampler, Interpolation, Argument Parse	Adam	LR= 1e - 4	Triplet loss	GPU	Cross-view; View Clothes; Carrying Spatial;Temporal
Zhang et al. (2019d)	Binary classifier	NA	NA	SGD Mean=0; SD= .01; M=.9	ED, Quintuplet Loss	CPU	Cross-gait
Takemura et al. (2017)	Softmax	L2 norm; LRN	NA	SGD	Contrastive, Triplet loss	CPU	Cross-view

Table 1 (continued)

Cite/year	Classification	Regularization augmentation	Optimizer	Training parameters	Loss function	GPU/CPU	Covariates handled
Zhang et al. (2019e)	Real/fake discriminator	BN	Euclidean distance	NA	Hard triplet loss	CPU	Cross-view
Zhang et al. (2019f)	fullyConnected layer	Data augmentation	NA	LR= .01 M= .9 WD= 0.0005	Softmax, MSE, Centre Loss	GPU	Cross-view
Wang and Yan (2020)	Softmax	Dropout technique	Backward propagation	SGD	Regression loss	CPU	Cross-view
Khan et al. (2020)	SVM	Weight decay; regularization	Mean square error	SGD LR= 0.0001	BP, LRL; Huber loss	GPU	Cross-view
Xu et al. (2020)	Triplet ranking loss	Local response normalization	Loss function	LR= 0.001 M= 0.9 WD= 0	Contrastive; Triplet loss; ED	CPU	Cross-view
Cai et al. (2019)	Horizontal pyramid pooling	NA	Adam	LR= 0.001	Triplet loss	GPU	Cross view Carrying
Feng et al. (2016)	Softmax	Dropout	NA	NA		CPU	Spatial Temporal; Pose
Haque et al. (2016)	Recurrent attention model	Gaussian noise; dropout= 0.5	Markov decision process	LR= 1×10^{-4} M= 0.9 WD= 5×10^{-4}	Cross entropy loss	CPU	Spatial Temporal
Uddin et al. (2019b)	Back propagation	NA	NA	NA	NA	CPU	Spatial Temporal
Li et al. (2017b)	Softmax	NA	NA	LR= .001 dec .1 at 10k iteration	NA	CPU	Spatial Temporal Cross-view Cross-subject
Castro (2017b)	Softmax	Dropout= 0.1; regularization	NA	Weight decay= 0.0005	NA	GPU	Spatial Temporal
Costilla-Reyes et al. (2018)	SVM	Batch Norm 1	RMSprop	SNE; LR= 0.001 dec by 10 when error plateaus	Log-likelihood loss	CPU	Spatial Temporal

Table 1 (continued)

Cite/year	Classification	Regularization augmentation	Optimizer	Training parameters	Loss function	GPU/CPU	Covariates handled
Karianakis et al. (2017)	Softmax	Dropout; Time as regularizer	Reinforcement learning objective	SGD M=0:5 LR= $3 * 10^{-4}$	Hybrid supervised loss	GPU	Occlusion; Pose View; Clothes
Batchuluun et al. (2018)	Softmax layer	Normalization layers	Adam	SGD M= 0.9 LR= 0.0001	Categorical cross entropy	GPU	Spatial Temporal
Liu et al. (2018)	Binary classification	L2 norm; Dropout= 0.7	Weight sharing	SGD LR= 10^{-4} M= .9; WD=.0005	Contrastive loss	CPU	Spatial Temporal
Tong et al. (2018)	Softmax	LRN; L2 norm	cross-entropy loss	LR= 0.001; M=0.9 WD= 0.005	Contrastive loss	CPU	Spatial Temporal
Karianakis (2018)	Sigmoid; Softmax	Dropout	Mean square error	SGD WD= 2∗ 10−4 LR= 3∗ 10−4; M=.5	Cross entropy loss	CPU	Depth Temporal
Battistone and Petrosino (2019)	supervised classification through LSTM	Dropout layer; Gaussian noise with $\sigma = 0.2$	NA	NA	NA	regular	Spatial Temporal
Wu et al. (2020)	Horizontal Pyramid Pooling	NA	NA	NA	NA	CPU	Spatial Temporal
Huynh-The et al. (2020)	Softmax	Normalizing	SGD	LR= .01 dropped 90% every 20 epochs	NA	CPU	Spatial Temporal Pose
Wu et al. (2015)	softmax layer; KNN	BN; Dropout=0.2	NA	LR= 0.1 M= 0.9 WD= 10^{-4}	Categorical cross entropy	GPU	Spatial View
Fan et al. (2020)	Euclidean distance	NA	Adam	LR= $1e^{-4}$ M= 0.9	Triplet loss	CPU	Spatial Temporal
Liu and Liu (2020)	Sigmoid	Dropout	Adam	LR= 0.0001	NA	GPU	Spatial Temporal Mine; View Clothes; Carrying
Rauf et al. (2016)	Softmax	Normalization layers	Gaussian distribution	LR=0.01 SD= .01 M= 0.9 Bias= 0 WD= 0.0005	Logistic regression loss	GPU	Gait

Table 1 (continued)

Cite/year	Classification	Regularization augmentation	Optimizer	Training parameters	Loss function	GPU/CPU	Covariates handled
McLaughlin et al. (2016)	Softmax	Unit variance; 0 MEAN norm	BP through time	LR= $1e^{-3}$	Cross-entropy loss	GPU	Gait Identity
Sokolova and Konushin (2017)	Nearest Neighbour	Batch Normalization	SGD	LR= 0.1	Triplet loss	GPU	Gait Identity
Marín-Jiménez et al. (2017)	Softmax; SVM; NN	Normalization; Dropout	SGD	LR= .01 M=.9; WD=.0005 SD= .01; Bias=0	Log loss; Regression loss	CPU	Gait Identity Gender Age
Nithyakani et al. (2019)	Softmax	NA	NA	NA	NA	GPU	Gait Identity
Carley et al. (2019)	Softmax	space-normalized	discrimination capability across viewpoint	NA	CCE; Hard triplet loss	CPU	Gait Identity
Song et al. (2019)	Back propagation	LRN; Dropout	Zero-mean Gaussian	SGD LR=.0001 SD=.01 Bias=0	Segmentation loss; Softmax	GPU	Gait Identity
Arshad et al. (2022)	Fine Tree; SVM; KNN	Skewness	Multi-Layer Perceptron	LR= 0.001	Cross entropy loss	CPU	End to end learning
Delgado-Escaño et al. (2020)	Softmax	Dropout= 40% L2regularization	NA	SGD LR= 0.01 M= .9 WD= 0.00005	NA	GPU	Occlusion Multi Gait
Jia et al. (2019)	Histogram distance	Spectral Norm Regularization	Adam	LR = 0.0002 M= 0.5, 0.999	Triplet loss	GPU	Adversarial
Wang et al. (2020b)	Softmax	Rotation, Flipping	Adam	Dynamic LR; M= 0.9, 0.999	Adversarial loss	GPU	Adversarial Super-resolution
He et al. (2020)	GaitSet (DNN)	Normalization	Adam	SGD	Cosine loss	CPU	Adversarial Robustness
Xue et al. (2020)	Linear Layer	Dropout= 0.5	Adam	LR= 10^{-5}, 10^{-4} M= 0.5 0.9	Margin Ratio Loss; SoftMax	GPU	Adversarial

Table 1 (continued)

Cite/year	Classification	Regularization augmentation	Optimizer	Training parameters	Loss function	GPU/CPU	Covariates handled
Xu et al. (2019)	Margine layer	Dropout=.5; BN; Linear normalization,	NA	SGD LR= 10^{-2} dec by 10^{-3} every 2500 epoc	Overall loss function; Margin loss	CPU	View; Multi walk Clothes; Crosswalk; Crossview
Zheng et al. (2016)	Re-id model	Lp−norm pooling	NA	NA	NA	CPU	Deformable View
Makihara et al. (2018)	Binary classification	Z-normalization	SGD	NA	Logistic regression	CPU	Deformable
Sakai et al. (2019)	Softmax	NA	NA	NA	NA	CPU	Speed
Zhu et al. (2018)	Adaptive late fusion	NA	NA	NA	Log-average miss	GPU	Gait; Face; Actions;
Alharthi et al. (2019)	Softmax	NA	SGD	NA	NA	GPU	Adversarial

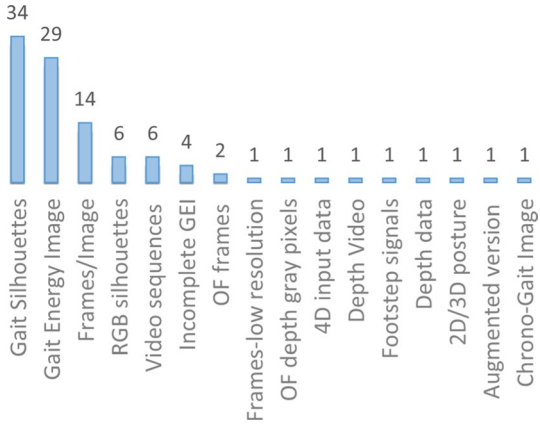


Fig. 3 Most adopted input techniques

Many times incomplete GEI is obtained due to occlusion, Babaee et al. (2018b) used incomplete GEI as input. Figure 3 depicts the most adopted input techniques.

Sometimes background subtraction is not precise when the silhouette color in the frame matches the background. Luo and Tjahjadi (2020) used RGB gait images; however, capturing features from such silhouettes results in feature loss in the areas where the silhouette color matches the background color. Luo et al. (2019) used gait silhouettes efficiently obtained after background subtraction. Uddin et al. (2019a) used low-resolution frames which are captured during video surveillance using closed-circuit television (CCTV). Castro et al. (2020) used optical flow images as input which depict the depth of an image and are included in the TUM dataset. In an indoor area, a kinect sensor can be used to collect gait data along with the depth data (Karianakis 2018). A motion control device based on augmented reality is centered on body segments captured by an action camera (Delgado-Escaño et al. 2020). Costilla-Reyes et al. (2018) used footstep signals obtained from footstep pressure to analyze various gait data deficiencies. Gait detection applications that utilize video-based 2-dimensional motion analysis are well known in a 2D position. It does not necessitate as much expertise as 3D systems, and the equipment needed is readily accessible and fairly priced due to the high computing expense. 3D techniques in surveillance have had limitations thus far, but Huynh-The et al. (2020) used 3D data to recognize gait. In different intelligent surveillance systems, the study of 4D dynamic with 4D input data scenarios with many walking pedestrians has piqued interest (Haque et al. 2016). The most used input technique is gait silhouettes (34%), followed by gait energy images, then frames/images, and after all, others.

2.2 Dataset description

A dataset is critical in implementing a gait recognition system which should contain all the real-time covariate conditions for training and testing purposes. More progress has been achieved in data gathering technology in recent years, and numerous factors that impact gait identification have been added for training. In this section, we examine gait datasets that have been in existence based on visual images and used in the cited literature. Figure 4 depicts the most adopted gait dataset. The most used gait dataset is CASIA B (63%), followed by OULP (31%), TUM-GAID 14%, and the rest.

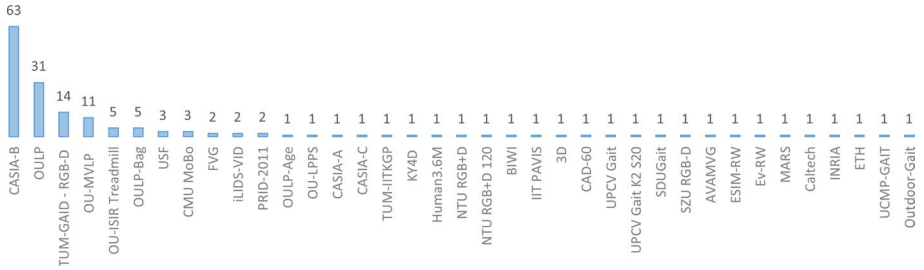


Fig. 4 Most adopted gait datasets

OU-ISIR treadmill B dataset (Makihara et al. 2012) comprises the subjects walking with 25 cameras surrounding the treadmill that take pictures at 60 FPS of 640 by 480 image size. Images of 68 subjects with clothing variations were taken from a side view while altering the combination of the attire. OU-ISIR large population dataset (Iwama et al. 2012) comprises subjects with two cameras surrounding the ground they are walking on. Dataset images contain silhouettes and are normalized to 88 by 128 as the camera took images at 30 fps from the size 640 by 480 pixels. The data set consists of images from over 4000 people of different ages. OU-ISIR large population dataset with age (Xu et al. 2017), comprises subjects from the age of 2 to 90 years. The 63,846 subjects walking on a track were captured by a camera that took images of 640 by 480 pixels at 30 fps. The GEI of the images taken was normalized to the size of 88 by 128. The dataset consists of both the testing and training data with 31,923 subjects.

OU-ISIR large population dataset with bag (Uddin et al. 2018) provides carry object details as a covariate. The dataset constitutes 62,528 subjects of age group in the range of 2 to 95 years. The subjects walked thrice in a straight course and the camera captured images of 1280 by 980 pixels at 25fps speed. The images were normalized and the background was subtracted to form the GEI of 88 by 128 pixels. OU-ISIR multi-view large population dataset (Takemura et al. 2018) is made by capturing the walking patterns of 10,307 subjects lying in the age group of 2 to 87 years. The images were captured from 14 different view angles and were 1280 by 980 pixels. Seven cameras were placed at 15° intervals that took images at 25fps. OU-ISIR multi-view large population dataset with pose sequence (An et al. 2020), is built upon OU-MVLP. Seven network cameras with a frame rate of 25 fps were used to capture the gait sequence of 10,307 subjects. The images were in RGB format and were of the dimensions 1280 by 980. In the OU-ISIR inertial sensor, dataset (Ngo et al. 2014), the sensors are positioned around the waist of 744 subjects of 2-78 years of age groups. The readings were taken on a flat surface as well as an up-slope and down-slope.

CASIA gait database A (Wang et al. 2003) consists of 12 image sequences of 20 subjects and provides four images of different view angles (0, 45, 90). There are a total of 19,139 images in the dataset. CASIA gait database B (Yu et al. 2006) provides clothing variation, carrying condition and view angle variation. The dataset captured 11 view angles and the dataset was built by capturing video sequences of 124 subjects. CASIA gait database C (Tan et al. 2006), provides infrared thermal images of 153 subjects walking at different walking paces captured at night. TUM-IITKGP gait database (Roy et al. 2011), provides occlusion sequences (long coat, hand in pocket, static occlusion, dynamic occlusion). TUM gait from audio, image and depth database (GAID) provides RGB video of 305 subjects in 3 different variations. It covers spatial-temporal covariate conditions. Human ID gait

challenge dataset (Sarkar et al. 2005), consists of a gait video of 122 subjects. The dataset provides cross view gait covariate with a mixture of 32-factor distinctions. CMU motion of body (MoBo) database (Gross and Cohn 2001) provides treadmill gait of 25 subjects and covers carrying, cross-view and speed covariates. The subjects were asked to walk in four different paces and six cameras were used to capture the images. KY4D gait database A (Iwashita et al. 2010), provides images and 3D models of 42 subjects walking in a straight course and covers clothing and pose covariates. Sixteen cameras were placed along the path to capture images that were further reconstructed using a visual hull approach.

Human3.6M: large scale dataset (Ionescu et al. 2014) consists of 3.6 million 3D images of 11 subjects with different poses under 17 distinct scenarios. The images were captured using four cameras of high resolutions. NTU RGB+D dataset (Shahroudy et al. 2016) consists of 56,880 images of 40 subjects in 60 action classes. The dataset covers view variation using three cameras. NTU RGB+D 120 dataset (Liu et al. 2019) consists of 8 million frames extracted from 114 thousand samples of videos of 106 subjects in distinct action classes. BIWI RGBD-ID dataset (Munaro et al. 2014) is constituted of 56 testing and 50 training progressions for 50 distinct subjects. The camera used captures videos at ten fps and the images are 1280 by 960 pixels. IIT PAVIS (Barbosa et al. 2012) consists of skeleton streams and a depth of 29 subjects. Microsoft kinect SDK was used to capture the frontal sequence of the subjects. MSR action 3D dataset (Imran et al. 2016) constitutes depth images of 20 actions captured using depth camera. This dataset covers spatial, temporal covariates and only one subject was used. Cornell Activity Datasets: CAD-60 (Sung et al. 2012) constitutes 57600 RGB-D video progressions of 60 subjects carrying out 12 distinct activities that are captured with the help of microsoft kinect sensor. UPCV1 (Kastaniotis et al. 2015; Khan et al. 2022) constitutes points of one subject taken indoor and considers spatial, temporal covariates. UPCV2 (Kastaniotis 2013) constitutes points of 18 subjects taken indoor and considers spatial-temporal and occlusion covariates.

SDUGait database (Wang et al. 2016) constitutes 2D and 3D data of 52 subjects. The dataset used two Kinect V2 to capture 21 joints and subtracted the foreground to form silhouette images. Dataset iLIDS Video reIDentification (iLIDS-VID) (Wang et al. 2014) constitutes 300 pedestrians captured with the help of two disjoint cameras in public space. This dataset considers clothing and views variation. Person Re-ID dataset (Hirzer et al. 2011) constitutes 1331 subject trajectories of view one and two combined, captured with the help of two cameras. SZU depth pedestrian dataset (Li et al. 2012) constitutes depth images of 4637 pedestrians along with 198 non-pedestrians captured using SwissRanger SR4000 camera in an indoor environment. AVA multi-view dataset for gait recognition (Spagnolo and Moeslund 2014) consists of gait sequences of 20 subjects in different trajectories. They considered view covariate. Human action recognition by 3D human skeletons (Vemulapalli et al. 2014) is considered pose covariate and constitutes 3D sequences of 10 subjects. The event-camera dataset (Mueggler et al. 2017) consists of 13 subjects and 26 sequences. Speed covariate is considered and the dataset was collected both indoor and outdoor. Caltech pedestrian detection benchmark (Dollar et al. 2011) or GM-ATCI rear-view pedestrians dataset constitutes over 200K pedestrians who mounted a standard automotive rear-view display camera for detection. The Inria aerial image labeling dataset (Dalal and Triggs 2005) covers a total of 810 kilometers square area and can be used in remote sensing. ETH dataset (Ess et al. 2008; Barth et al. 2015) consists of pedestrians walking that have been captured with the help of a stereo rig mounted on a car that captures video at 13–14 fps. Seely et al. offer a 3D dataset that included 103 people and had 1030 samples total, of which only 1005 were considered legitimate (Seely et al. 2008).

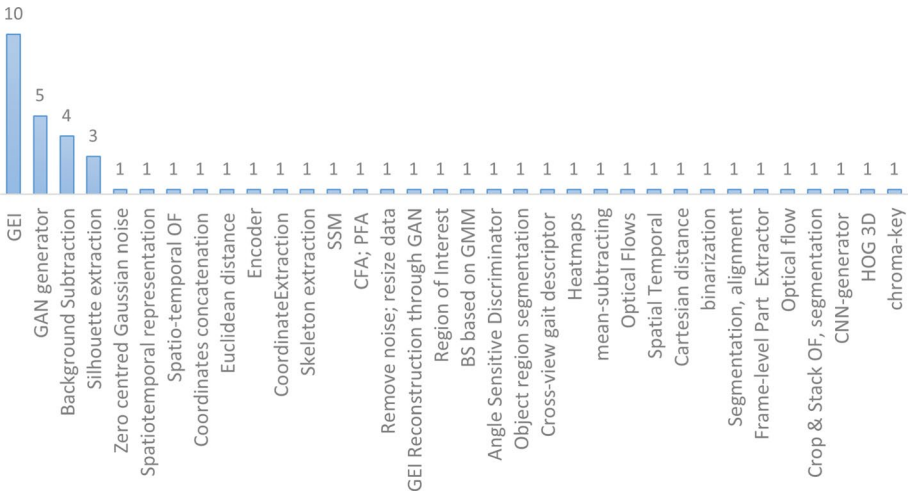


Fig. 5 Most adopted features and data preprocessing techniques

In fourteen invalid samples out of twenty-five, the clipping came from when the subject was located outside the reconstruction region. The use of three-dimensional data allows silhouettes to be synthesized from any arbitrary perspective, even if a camera does not immediately view the viewpoint. This is possible because silhouettes may be represented in three dimensions. Hadid et al. explored how clothes might impersonate a target, indicating that such attacks can exploit major vulnerabilities (Hadid et al. 2012).

2.3 Feature selection and data preprocessing techniques

After the input is taken in the gait pipeline, it is passed to the next stage of the gait pipeline for preprocessing and feature selection. Figure 5 depicts the most adopted preprocessing techniques in gait recognition.

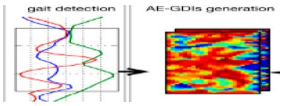
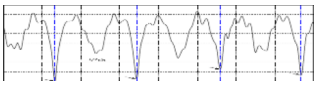
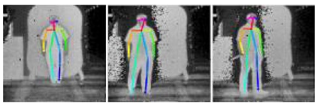
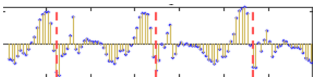
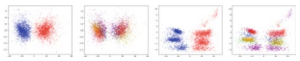
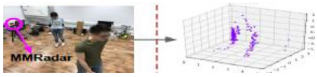
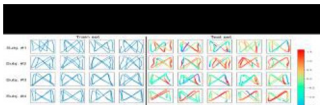
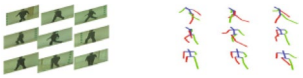
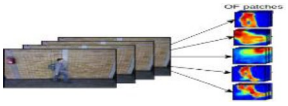
The most used preprocessing technique is GEI (10%), followed by GAN (5%), background subtraction (4%) and the rest. Table 2 shows the various techniques used by the researchers for data preprocessing and feature selection.

The primary and crucial stage after collecting gait videos is to identify and monitor the target subject in the video. The selection of data preprocessing techniques plays a vital role in the recognition rate and, thus, becomes a paramount concern in selecting the best one. In gait recognition, while doing data preprocessing, some techniques are frequently used, like background modeling, region of interest finding, noise removal, resizing the image, and silhouette extraction.

2.4 Deep learning techniques

Various deep learning techniques are used to recognize gait. We provide a categorical overview of deep learning techniques in Fig. 6 and brief descriptions of each approach used to recognize gait. The four deep learning strategies are supervised learning, unsupervised

Table 2 Data preprocessing and feature selection techniques

Cite	Data preprocessing	Image
Xia et al. (2019), Castro et al. (2017)	Cycles Extraction, normalized	
Tong et al. (2017)	Angle embedded dynamic Image	
Li et al. (2019)	Discriminant, class-invariant features	
Ling et al (2019)	Outlier detection	
Yu et al. (2019)	Gait Cycle Segmentation	
Yu et al. (2017)	CNN	
Cheheb et al. (2018)	Kalman filter	
Thapar et al. (2019)	Low pass filtered at 30 Hz; down sampled to 50 Hz	
Castro et al. (2020)	Event noise cancellation	
Shiraga et al. (2016)	3D Pose estimation	
Zhang et al. (2017)	Motion map computation	
Li et al. (2017)	Morphological filters	
Jia et al. (2017)	Synthesized 3D gait model	
Thapar et al. (2018)	Normalized 2D Pose	

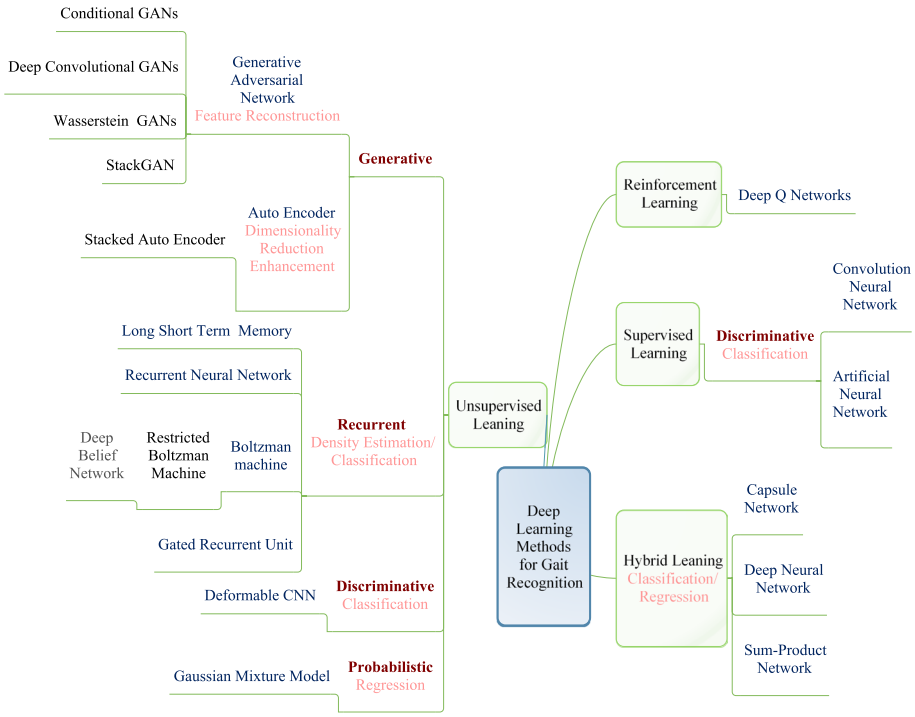


Fig. 6 Categorical review of deep learning techniques

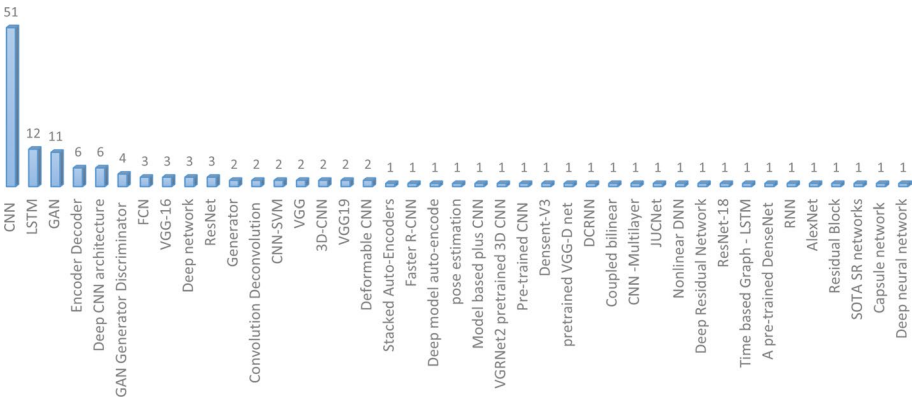


Fig. 7 Most used deep learning techniques

learning, hybrid learning, and reinforcement learning. There are six types of networks: feedforward, generative adversarial network, recurrent network, hybrid learning, discriminative and user-defined network. The most used deep learning techniques are CNN (51%), followed by LSTM (12%), GAN (11%) and the rest are depicted in Fig. 7.

2.4.1 Feedforward network

When a machine learning algorithm utilizes several layers of nodes to generate high-level functions from input data, it is called a deep neural network. It entails translating data into a more abstract and artistic aspect (Zhu et al. 2018). Traditional feedforward networks use multi-layer perceptrons; however, CNN has recently exceeded them. To translate image input to a vector output, convolutional neural networks (CNNs) are utilized. They have become the de facto norm for any prediction problem that requires image data as an input (Wang et al. 2020a). CNNs are a powerful paradigm for learning representations for volumetric data since they take a 3D volume or a series of 2D frames as input (Thapar et al. 2018). Fully connected networks (FCNs) lack dense layers (as do traditional CNNs), relying instead on 1x1 convolutions to perform the role of fully connected layers (dense layers) (Babaee et al. 2018b). Deconvolution transposes the convolution layer, which is used to sharpen features affected by fast motion capture (Babaee et al. 2019b).

VGRNet2 is a 3D CNN network and instead of beginning from scratch, it has been pre-trained to solve its clothing covariate problem (Mehmood et al. 2020). Densent-V3 is a commonly used image recognition model that uses convolutional neural networks to allow faster computation and deeper networks by reducing dimensionality with stacked 11 convolutions. The modules were created to address computing cost and overfitting problems (Wu et al. 2015). VGG16 (also known as OxfordNet) is a convolutional neural network model capable of adapting to a wide range of input sizes (Sun and Liu 2018), and VGG-19 is a 19-layer deep convolutional neural network (Li et al. 2017b). The depth and amount of completely linked nodes of a qualified VGG-D network, VGG16, is over 533MB which helps to solve view variation covariate conditions (Zhang et al. 2017). Unfortunately, VGGNet has two significant flaws: it is incredibly slow to learn and develop; second, the network architecture weights are very high. Many deep learning gait recognition problems use VGG; however, narrower network architectures are often preferred (such as SqueezeNet, GoogLeNet) for cross-view covariate conditions (Zhang et al. 2019e). The design of AlexNet is made up of eight layers: There are five convolutional layers and three linked layers (Arshad et al. 2022).

2.4.2 Generative adversarial network

A generative adversarial network (GAN) is an analytic framework that matches two neural networks against each other to produce fresh, simulated occurrences of data that would compete for actual data. They are commonly used in video and photogeneration, as well as in gait recognition to generate the occluded images and enable the model to learn in an unsupervised manner (Li et al. 2019a). Over time, the generator network gains the capacity to produce believable artificial data, such as fake images. It utilizes discriminator feedback to gradually improve its accuracy until the discriminator cannot distinguish between the output and actual results (Babaee et al. 2019a). In a GAN, the discriminator is a classifier. It tries to differentiate between real data and data produced by the generator. It can use any network infrastructure that is suitable for data classification (Xia et al. 2019).

2.4.3 Recurrent network

In recurrent networks, nodes are linked in a network that is directed by a temporal series, thus preserving the temporal features. This feature of recurrent networks allows them to act in a time-complex way, and temporal data can be easily captured. McLaughlin et al. (2016). In contrast to traditional feedforward neural networks, LSTM is a recurrent neural network (RNN) with input connections. It can deal with individual data points as well as whole data sequences (Li et al. 2019a). The most traditional time-dependent graph LSTM system forecasting with graph convolution neural network is based on statistical and autoregressive approaches (Battistone and Petrosino 2019). A residual neural network is a form of artificial neural network based on pyramidal cell structures in the cerebral cortex. Skip links, or shortcuts, are used by residual neural networks to hop through certain layers (Yang et al. 2019). Residual blocks are a form of highway network that does not have any gate in its skip connections. Residual blocks, in essence, enable memory (or information) to move from the first to the last layers (Jia et al. 2019).

An autoencoder is a neural network made up of several progressively increasing, sparsely valued layers, each of which has an output connected to the input of the next lower-valued layer by a hidden connection. Auto-encoders are a series of encoders and decoders connected in a network. The encoder converts the incoming data into intermediate representations known as latent variables. The decoder uses these latent variables to reconstruct the input data. However, verifying that the network does not memorize the data is necessary. Nevertheless, it is vital to identify features that characterize it accurately. Auto-encoders include contractive, sparse, variational and denoising auto-encoders. Encoder-decoder, also known as seq2seq, is a recurrent neural network programmed to solve sequence-to-sequence problems. Since parameters in the input and output sequences will differ, sequence-to-sequence prediction problems are challenging to solve (Babaee et al. 2019a). A stacked autoencoder increases deep learning efficiency as noisy autoencoders are inserted in the layers (Yeoh et al. 2017). The spatial dependence is captured using bidirectional paths on the line. The temporal dependency is captured using an encoder-decoder architecture with scheduled sampling in the diffusion convolutional recurrent neural network (Deng et al. 2020).

2.4.4 Hybrid learning

The capsule network (CapsNet) is the newest addition to the feedforward networks, explicitly designed to address inherent discriminator constraints of the CNN. Current DL research is focused on capsules, which are learned via a dynamic routing technique. Even while CapsNets are still in their early stages, they have already shown their power to change the DL landscape. Nonlinear deep neural networks have nonlinear activation layers, which gives them their nonlinearity. To help hierarchical model relationships capsule neural network must be used. The aim is to imitate biological neural organization as nearly as possible (Xu et al. 2019). The neural network and the amount of training it receives determine the mechanism for relating the input and output (Khan et al. 2020). Faster R-CNN is fed into a CNN named the region prediction network (RPN). It considers a larger number of potential regions than the initial R-CNN algorithm. It employs a quick deep learning approach to predict which regions are more likely to contain subjects of interest (Sun and Liu 2018).

2.4.5 Discriminative and user defined networks

Deformable convolution and deformable RoI pooling are built on the concept of augmenting the spatial sampling positions with additional offsets. Knowing the offsets from goal tasks is given more importance (Zheng et al. 2016). Coupled bilinear produces aligned gait matrix characteristics for two viewpoints using two sets of bilinear transformation matrices while keeping the spatial structural detail of the original GEI. It will iteratively optimize the inter-class distance metric to the intra-class distance metric to discover the ideal matrix subspace where the gait energy images surrounding views are matched in horizontal and vertical coordinates (Ben et al. 2019). JUCNet (Joint Unique-gait and Cross-gait Network) is a network that combines the advantages of unique and cross gait conditions, resulting in a significant increase in overall accuracy (Zhang et al. 2019d). A pose estimation network is a computer vision strategy for predicting and tracking the position of a person or object (Liao et al. 2017). 2D Skeleton pose estimation uses GPU acceleration to achieve low-latency joint real-time object identification and 2D main point pose estimation with high precision (Thapar et al. 2019). Image super-resolution (SR) strategies use deep back projection networks to recreate a higher-resolution image, or series (Wang et al. 2020b).

2.5 Feature extraction and representation

When dealing with huge amounts of raw data, feature extraction is a method that divides the data into smaller groups that may be processed more efficiently. Table 3 provides a detailed overview of the various methods used in deep learning frameworks with different feature extraction and representation techniques explored. Figure 8 depicts the most adopted feature extraction techniques in gait recognition. The most used feature extraction technique ratio is CNN (62%), followed by the encoder (10%) and others.

2.6 Feature reduction and transformation

Feature reduction and transformation can be accomplished in various ways. The most adopted feature reduction and transformation techniques are PCA (10%), followed by the discriminator (8%), decoder (5%), LDA (4%), and others. Figure 9 depicts the most adopted feature reduction and transformation techniques in gait recognition.

A discriminator is nothing more than a classifier, and it attempts to tell the difference between real data and data produced by the generator (Wang et al. 2019b). When accompanied by a typical convolutional layer in a generative model, an up-sampling layer is a basic layer with no weights that double the dimensions of data (Babaee et al. 2019b). The gait characteristics in the probe viewing angle are transformed to those in the gallery viewing angles using a view transformation model (He et al. 2019). Techniques including flipping, rotating, zooming, clipping, translating, and adding noise also increase with the amount of data accessible. Data augmentation is a method for creating fresh training data using current training data artificially (Wang et al. 2019a). Sparse representation aims to reflect signals with as few significant coefficients as possible. This is effective for several uses, including compression (Xia et al. 2019). As the name suggests, Dimensionality reduction methods minimize the number of measurements (i.e., variables) in a dataset while preserving as much data as possible (Alotaibi

Table 3 Feature extraction and representation details

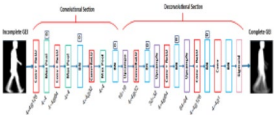
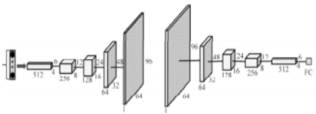
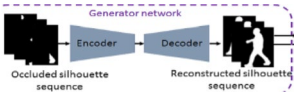
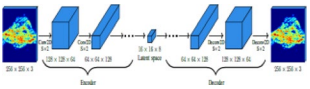
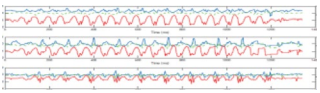
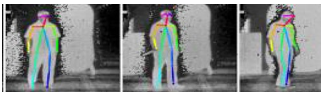
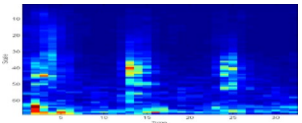
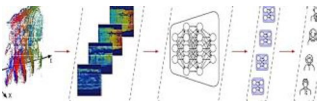
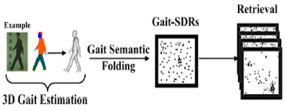
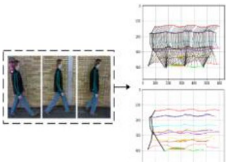
Cite	Technique	Image
Babaee et al (2018), Babaee et al. (2018), Xia et al. (2019), Babaee et al. (2019), Li et al (2019), Das et al. (2019), He et al. (2019), Uddin et al. (2019), Wang et al. (2019), Yeoh et al. (2016), Alotaibi and Mahmood (2017), Castro et al. (2017), Tong et al. (2017), Yang (2019), Ling (2019), Castro et al. (2019) , Li et al. (2020), Luo and Tjahjadi (2020) , Li et al. (2020), Yu et al. (2017), Yu e al. (2019), Alotaibi and Mahmood (2017) , Yu et al. (2017), Yao et al. (2018), Cheheb et al. (2018), Zhang et al. (2019), Thapar et al. (2019), Mehmood et al. (2020), Swee (2014), Shiraga et al. (2016), Wolf et al. (2016), Li et al. (2017), Zhang et al. (2019), Deng (2020),	CNNs	
	Encoder	
Babaee et al. (2019)	Encoder, Decoder	
Yeoh et al. (2017)	Convolutional autoencoder	
Sun and Liu (2018)	Quasi-periodic signals	
Yan et al. (2015)	3D Joint location estimator	
Liao et al. (2019)	Scattering transform	
Castro et al. (2020)	CNN with Residual Block	
Zhang et al. (2017)	OF, average pooling	
Jia et al. (2017)	3D Gait Estimation	

Table 3 (continued)

Cite	Technique	Image
Thapar et al. (2018)	Spatial temporal feature	
Zhang et al. (2019)	Reconstructed trajectories and encode feature embeddings	

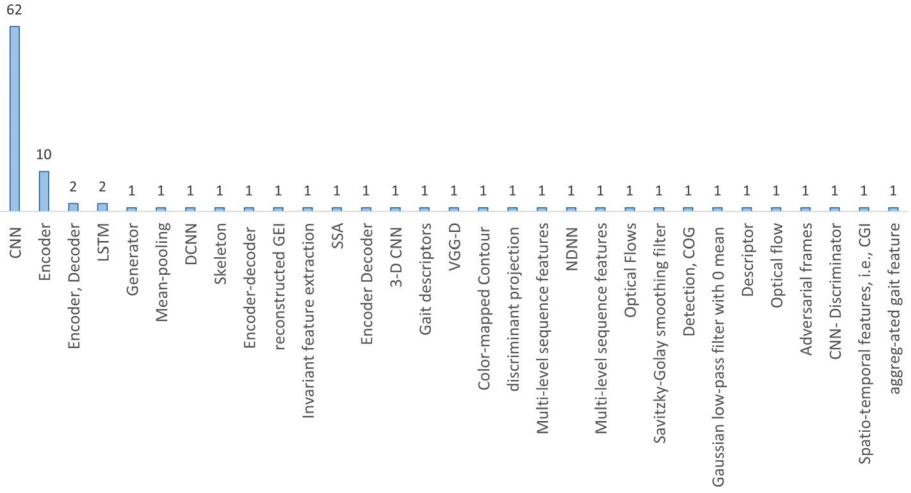


Fig. 8 Most adopted feature extraction techniques

and Mahmood 2017). PCA is a way to reduce the dimensions of datasets, increasing interpretability and thus decreasing knowledge loss. It accomplishes this by generating new uncorrelated variables that optimize variance in a sequential manner (Zhang et al. 2017). Linear discriminant analysis (LDA) is a method for reducing dimensionality (Alotaibi and Mahmood 2017). Auto-encoders can convert data input to a lowered or encoded output to protect data or reduce storage space reduction. The hidden parameter would be converted into the output sequence by the decoder (Li et al. 2020b).

A variety of filters in pooling layers, such as the horizontal pooling layer, are used in deep learning networks (Fan et al. 2020). The pooling function reduces the dimension of the feature matrix, which is extracted by the convolution layer (Castro et al. 2020). The color image is processed using the YCbCr transformation. The result is then processed using a neural network on the Y channel to produce a grey image filled with color details to produce the color effect image (Mehmood et al. 2020). The rectifier feature is used to improve non-linearity in gait images, which is why pooling by a rectifier is used (Jia et al. 2017). Linear interpolation is a curve-fitting technique that employs linear polynomials to

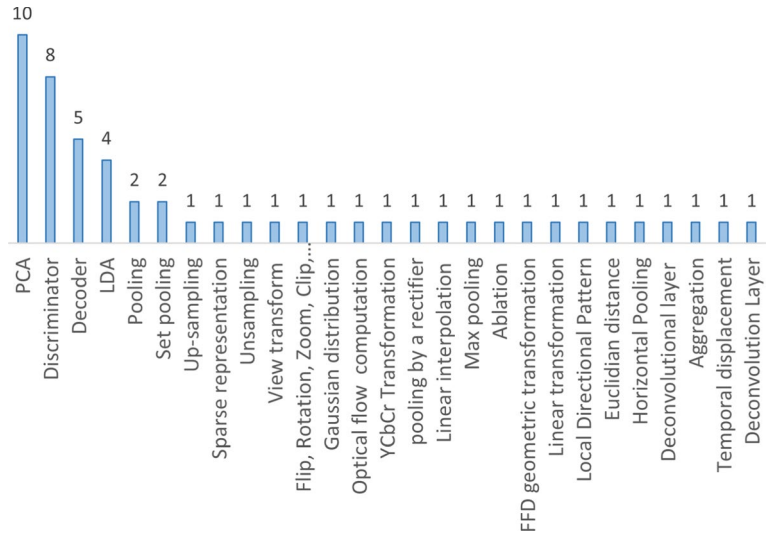


Fig. 9 Most adopted feature reduction techniques

create class labels well within the context of a distinct collection of existing data points (Linda et al. 2020). Maximum pooling, also known as max pooling, is a pooling process that determines the maximum value in the function map. In the case of average pooling, the findings are downsampled and highlight the most present feature in the patch rather than the average existence of the feature (Chao et al. 2019).

Geometric transformations of free form in the deformation network help to infer the detailed shape of instance deformation from its canonical shape mask (Xu et al. 2020). A linear transformation is a process of preserving each vector space underlying linear structure (Haque et al. 2016). The term "ablation analysis" refers to the process of eliminating a "function" of a model or algorithm to see if it affects the results (Zhang et al. 2019d). Local directional patterns are used to track gait edges in greyscale images (Uddin et al. 2019b). The propagation of visible movement velocities of a brightness pattern in a gait image is often known as optical flow (Hawas et al. 2019a). Euclidean distance is used to locate the sample by assigning weights nearest to the desired sample and belonging to the same class. When several samples have the same distance to the reference sample, the first one detected is used (Batchuluun et al. 2018). Through segmented image generation, such as semantic segmentation, the deconvolution layer is used to conduct computations from the output to the input layer (Song et al. 2019; Khan et al. 2022). Aggregation is a way of reducing many variables into a few numerical values or figures (Delgado-Escaño et al. 2020). A distribution is formed by a sample of results and the gaussian distribution, also known as the normal distribution, takes the likelihood for every observation from the sample space using parameterized mathematical function (Yao et al. 2018).

2.7 Normalization, augmentation, regularization

Normalization is a method of decomposing tables to remove data duplication and unwanted features such as insertion, updation, and deletion anomalies. It is a multi-step method that

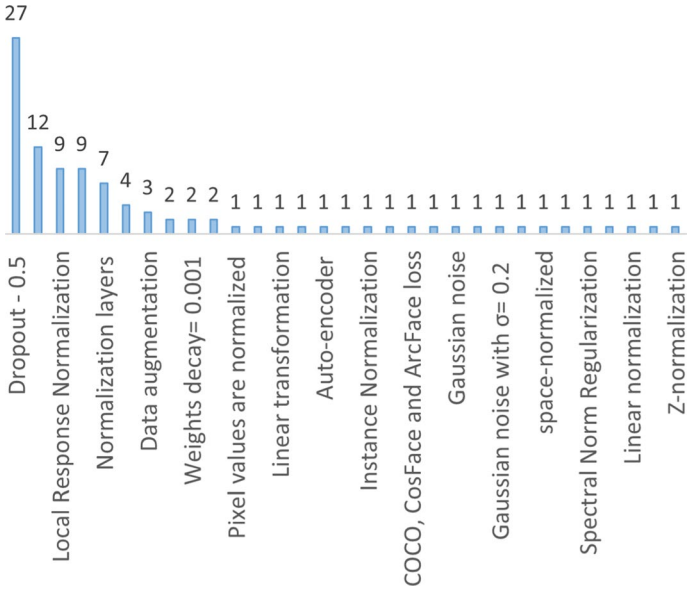


Fig. 10 Most adopted regularization, normalization, augmentation

converts data from connection tables into tabular form, thus eliminating redundant data. The volume of data accessible also increases the efficiency of deep neural networks. The most used regularization and augmentation technique is dropout (27%), followed by batch normalization(12%), L2 normalization (9%), LRN (9%), and others. Figure 10 depicts the most adopted normalization, augmentation, and regularization techniques in gait recognition.

Normalization aims to bring an image into an intuitive range and is used when the data is linear (Xu et al. 2019). In a network, the allocation of inputs to the layers changes because of normalization layers (He et al. 2020). Normalization is commonly used in unit variance and average zero mean conditions (McLaughlin et al. 2016). The space normalized function removes leading and trailing whitespace from a string, replaces whitespace sequences with a single space, and returns the output (Carley et al. 2019). The method of linear normalization alters the spectrum of pixel values. Batch normalization is a training strategy for very deep neural networks that standardize the inputs of all mini-batches. Consequently, the learning phase is stabilized, and the number of training epochs required to train deep neural networks is greatly reduced (Zhang et al. 2019a). By normalizing over local input areas, the local response normalization layer conducts a type of "lateral inhibition" (Shiraga et al. 2016). The instance normalization mean and variance is measured for all spatial dimensions, i.e., for each subject in every image pixel (Swee et al. 2014).

Dropout is a training method in which neurons are rejected randomly and "disappear" at random. This implies that their impact on downstream neuron activity during the forward pass is eliminated, and any weight changes are not communicated to the neuron during the backward pass (Das et al. 2019). In linear transformation, the linear structure is maintained from one vector space to another by preserving each vector space structure. A linear operator or map is another name for a linear transformation (Yeoh et al. 2017). L1 norm is the number of magnitudes of the vectors in space. The sum of absolute differences of

components in vectors is the most natural way of measuring the distance between vectors. All the components of the vectors are evenly weighted in this norm (Li et al. 2019b). L2 normalization is also known as least squares, and it changes the dataset values such that the number of squares in each row is always greater than one (Castro et al. 2017a). Lp-norm pooling attempts to learn the pooling parameters discriminatively (Zheng et al. 2016; Giorgi et al. 2021). Z-normalization is a strategy of normalizing data that avoids outlier issues (Makihara et al. 2018). The comparison of a time series with itself at a particular time is known as normalized autocorrelation (Li et al. 2017a).

Data augmentation is a method for artificially creating extensive training data using current training data (Yan et al. 2015), thus modifying images in the training set that the model is likely to see. Rotation or flipping is the data augmentation technique to transform data for deep learning models (Wang et al. 2020b). Duplication and rotation are the data transform techniques used (Alotaibi and Mahmood 2017) to augment the data. In (Li et al. 2020a), translation is used as a data augmentation technique. Auto-encoder is used to generate the data (Yu et al. 2017); auto-encoders are composed of an encoder and decoder. The encoder extracts the features, and the decoder decodes them. A statistical noise with a probability density function equal to the regular distribution is known as gaussian noise (Haque et al. 2016). The 2D gaussian function is used to estimate the intensity distribution provided by a point source (Linda et al. 2020). Argument, triplet sampling, and interpolation in a fully linked layer get the data points by parsing several centers, and each gait image is allocated to one of them. Modeling intra-class variation in real-world gait datasets is easier with it (Chao et al. 2019).

Regularization is a method for tuning the process that involves inserting an extra penalty word into the error function. The additional term regulates the excessively fluctuating function, preventing the coefficients from taking extreme values. Weight decay is a regularization technique that puts weight to cost function. If no additional update is scheduled, weight decay is a notion in the weight update rule that permits the weights to decay exponentially to zero (Hawas et al. 2019b). Weight decay with the value of 0.001 is a parameter that controls how much an updating phase affects the current value of the weight. Losses like COCO apply a cosine margin penalty to the goal logit and are, therefore, simpler to practice (Zhang et al. 2019d). L2 regularization makes the weights minimal but not empty, and it produces a non-sparse solution (Hayfron-Acquah et al. 2003). A data collection's skewness is a numerical measure of how often it deviates from the normal curve. If the data distribution means is less than the mode, there may be more graphed dots on the left than on the right, resulting in a negative skew distribution (Arshad et al. 2022). Norm spectral regularization is a technique for reducing the resilience of a system to perturbation (Jia et al. 2019). Time as a regularizer modifies the learning algorithm slightly to improve the model generalization. As a result, the success of the model on previously unseen data increases (Karianakis et al. 2017).

2.8 Activation function

In deep learning, a node's activation function influences its output by providing input or a collection of inputs. A typical integrated circuit may be considered a digital network of activation functions that can be turned on or off based on the input. The nonlinear activation mechanism enables the model to construct complex mappings between the inputs and outputs of the network, which are required for studying and modeling complex data such as images, video, audio, and nonlinear or high-dimensional datasets (Xia et al. 2019).

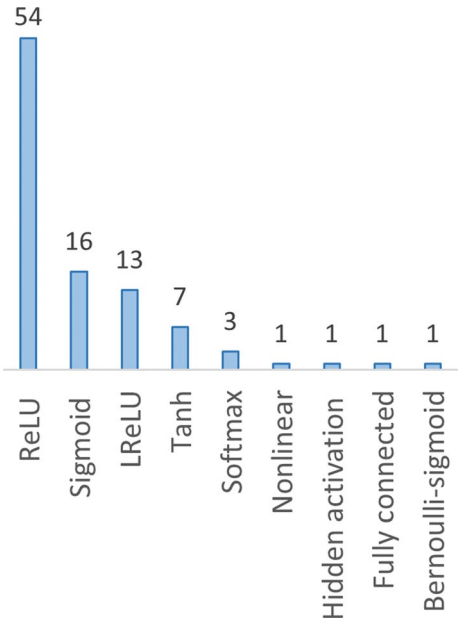


Fig. 11 Most adopted activation function

The most adopted activation function is ReLU (54%), followed by sigmoid (16%), LReLU (13%), and others. Figure 11 depicts the most adopted activation functions used in the deep learning pipeline to recognize gait.

An activation function with a rectified linear activation (ReLU) is a piecewise linear function that outputs the input directly if it is positive and zero otherwise. As a result of the corrected linear activation function, the problem of vanishing gradients is resolved, allowing models to learn more quickly and perform better (Nithyakani et al. 2019; Tieu et al. 2017; Saber et al. 2021). Leaky Rectified Linear Activation or LReLU derivative is one in the positive portion and a slight fraction in the negative part (Yu et al. 2017). The sigmoid activation function, commonly known as the logistic function, has long been used in neural networks to represent the activity of neurons. Using this function, the input is transformed to an integer in the range of 0.0 to 1.0 (Alotaibi and Mahmood 2017). Tanh is derived from (-1 to 1). The benefit is that in the tanh graph, negative inputs will be mapped highly negative, and zero inputs will be mapped near zero (Hayfron-Acquah et al. 2003). The softmax function is the activation function in deep network models' output layer that replicates a multinomial probability distribution. For every node in the final layer, softmax activation produces a single value (Zhang et al. 2019d). Hidden activation functions convert the obtained feedback to hold values within a manageable range (Liao et al. 2017). Each output dimension depends on each input dimension in a fully connected layer (Liu et al. 2018).

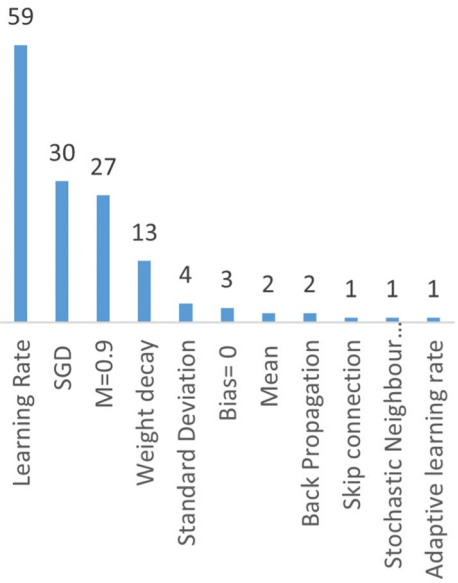


Fig. 12 Most adopted training parameters

2.9 Training parameters

To configure the training and hyper-parameters, one needs much practice and a lot of trial and error. Setting hyper-parameters such as learning rate, batch size, momentum, and weight decay requires time and effort. Let us start by defining these hyper-parameters. These hyper-parameters act as knobs that may be adjusted during the model's development. The most used training parameters are learning rate (59%), followed by SGD (30%), momentum (27%), weight decay (13%), and others. Figure 12 depicts the most adopted training parameters in gait recognition.

The learning rate is a hyperparameter that indicates how much the algorithm may adapt each time the model weights are adjusted in response to the predicted error (Xue et al. 2020). Adaptive learning rate aims to reduce the learning rate until the output of the model reaches a peak, such as by a factor of two or an order of magnitude (Alotaibi and Mahmood 2017). The gradient descent algorithm uses the expression "momentum" (M=0.9). Gradient descent is an optimization algorithm that finds the path of the steepest slope in its current state and changes it by going in that direction (Wolf et al. 2016). Weight decay is a regularization procedure that involves applying a minor penalty to the loss parameter, normally the L2 norm of the weights (Wolf et al. 2016). Stochastic gradient descent (SGD) is a quick and easy method for fitting linear classifiers and regressors to convex loss functions (Tong et al. 2017; Zheng et al. 2022).

Deep architectures with skip connections skip several layers in the neural network and feed the output of one layer as a reference to the next layers (Das et al. 2019).

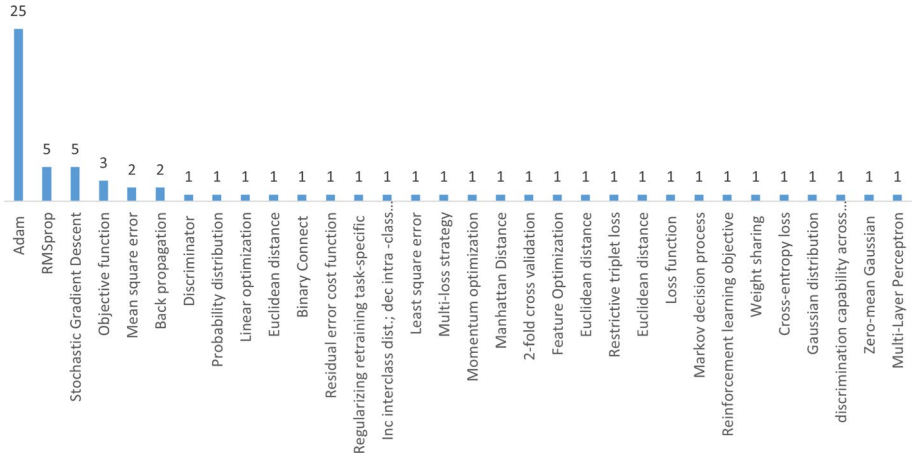


Fig. 13 Most adopted optimizer details

Backpropagation is a process that uses an optimization algorithm such as gradient descent (Swee et al. 2014). The mean function finds the average value of the variables (Zhang et al. 2019d). The standard deviation is a calculation of a group of the variance value. A low standard deviation means that the values are like the mean value of the set, while a high standard deviation indicates that the values are distributed across a larger spectrum (Rauf et al. 2016). Stochastic neighbor embedding (SNE) is a nonlinear, unsupervised technique for discovering and visualizing high-dimensional data. SNE provides a sense of how data is organized in a high-dimensional space (Costilla-Reyes et al. 2018). The activation feature may be moved to the left or right using the bias value to best match the results (Rauf et al. 2016; Marín-Jiménez et al. 2017; Song et al. 2019). Algorithms with a low variance but a large bias are usually simpler. Low bias with high variance algorithms is more challenging in implementation.

2.10 Optimizer

An optimizer is a strategy or technique for updating numerous parameters to minimize loss with little effort. The most used optimizer is adam (25%), followed by SGD (5%), RMSprop (5%), and others. Figure 13 depicts the most adopted Optimizers in gait recognition.

Adam optimizer includes characteristics of both RMSProp and AdaGrad optimizers to solve noisy problems with sparse gradients (Zhang et al. 2019b). RMSprop uses the magnitude of recent gradients to normalize the gradients. Costilla-Reyes et al. (2018) divide the present gradient by a moving average over the root mean squared gradients. He et al. (2019) used a discriminator optimizer with steps of the gradients accumulated from both the all-real and all-fake batches of the discriminator. Linear optimization aims to find the values of the variables that maximize or minimize the objective function (Li et al. 2019b). Unknown variables are represented using probability distribution functions (Tong et al. 2017).

In a huge gait dataset, euclidean distance can be used to efficiently locate several global optima on the questing landscape over certain commonly utilized multimodal optimization test functions (Ling 2019; Wu et al. 2016). The learned parameters are

restricted to 0 or 1 in a binary connect optimizer (Castro et al. 2019). Iteratively running the model and comparing predicted forecasts to ground reality is used to measure the residual error cost function (Luo and Tjahjadi 2020). The method of applying details to a task-specific optimizer to avoid overfitting is known as regularizing retraining (Yan et al. 2015). Yu et al. (2019) used an optimizer to improve interclass distance and decrease the intraclass distance. To evaluate and visualize results, use least square error, where the objective (error) function is a quadratic function of a parameter(s) being optimized (Yu et al. 2017). In a multi-loss strategy, a model must balance several parameters, and the usual solution is to reduce a loss function that is the weighted total of the parameters (Liao et al. 2017). Momentum optimization is a technique that aids in the acceleration of SGD in the desired direction (Yao et al. 2018). In multiobjective optimization issues, the manhattan distance approach to multiple criterion decision making is used (Mehmood et al. 2020). 2-fold cross-validation is a method for evaluating predictive models that divides the initial dataset into a train and test set to learn and validate the model (Li et al. 2017a).

Feature optimization reduces feature space by picking and utilizing functions specific to answers and readily distinguishable from other samples (Jia et al. 2017). The objective function is a real-valued function whose value must be reduced or maximized over all possible options. It is conceivable that there are many ideal solutions; in reality, there may be an infinite number of them (Hu et al. 2048). Restrictive triplet loss is a loss function that compares a baseline input to a positive and negative input. The distance between the baseline input and the positive input is kept as small as possible, whereas the gap between the baseline input and the negative input is kept as large as possible (Huang et al. 2018). Stochastic gradient descent is computing the derivative from each training data instance and immediately calculating the change (Marín-Jiménez et al. 2017). The mean square error (MSE) is used to see how accurate predictions or projections are to real values. MSE is a model validation metric for regression models, and the lower the value, the better the fit (Khan et al. 2020). The squared-error loss is a kind of loss function that grows quadratically with the difference and is used in estimators such as linear regression, impartial statistics estimation, and several areas of machine learning (Xu et al. 2020).

A discrete time stochastic management loop is the markov decision process. It offers a statistical basis for modeling decision making in contexts where results are partly unpredictable and partly regulated by a decision maker (Haque et al. 2016). Reinforcement learning aims to choose the best-known response for each given state, which necessitates ranking and assigns relative values to the behaviors (Karianakis et al. 2017). Weight sharing develops much more powerful optimization algorithms that enable neural networks to manage problems more effectively, allowing them to learn faster and perform better (Liu et al. 2018). The efficiency of a classifier whose output is a probable number between 0 and 1 is measured by cross-entropy loss. Cross-entropy loss rises as the expected likelihood differs from the actual label (Tong et al. 2018). Gaussian distribution is based on an inferior solution repair method to modify the ill-shaped distribution of the dataset (Rauf et al. 2016). Discrimination capability (Carley et al. 2019)across viewpoints is used to determine the most discriminating gait features. Zero-mean gaussian is used in (Song et al. 2019), and the value of standard deviation is one. Backpropagation is a technique to backpropagate errors while training the model (McLaughlin et al. 2016) by changing the parameters (weights and biases). Multi-layer perceptron is used in Arshad et al. (2022), which

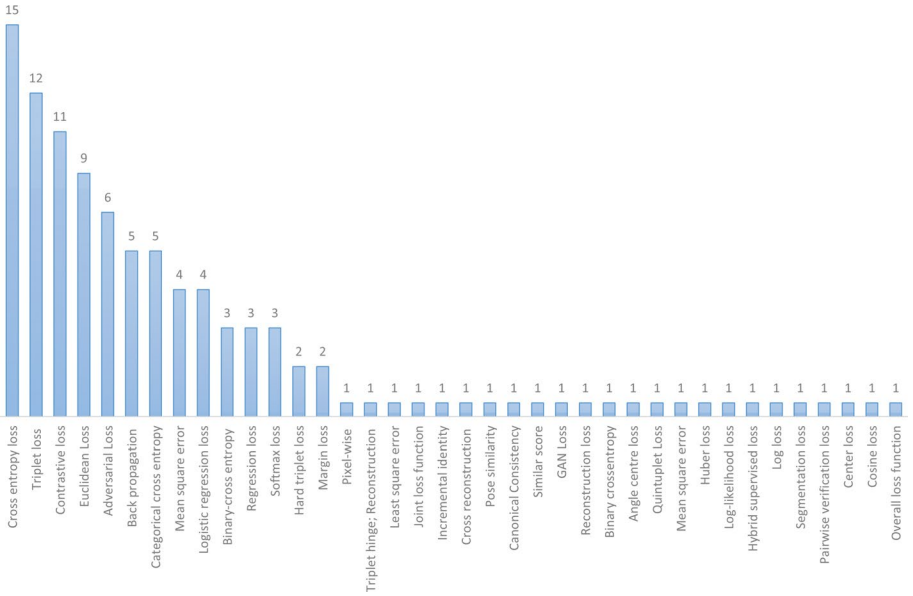


Fig. 14 Most adopted loss function details

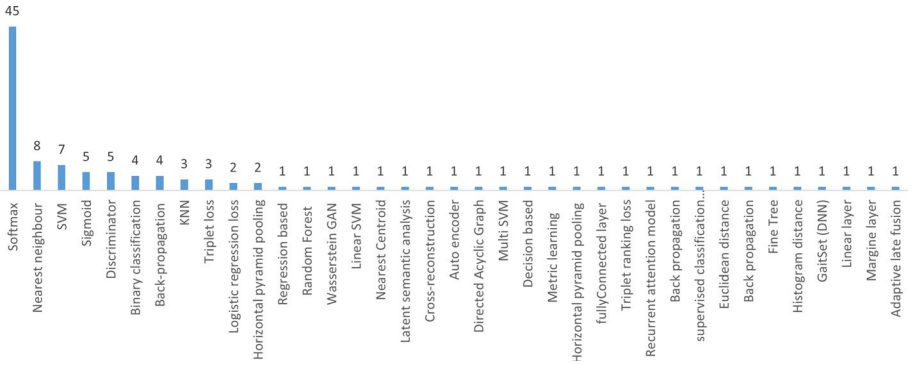


Fig. 15 Most adopted classification functions

uses backpropagation to train the model. Proper weight adjustment in backpropagation reduces error rates and improves model tuning by increasing generalization.

2.11 Loss function

A loss function, also known as a cost function, is a mathematical function that converts an occurrence or the values of one or more variables into a real number to depict the "cost" associated with it. A loss function is minimized in an optimization problem. Figure 14 represents various loss functions used in literature.

Figure 14 depicts the most adopted loss functions in gait recognition. The most used loss function is cross-entropy loss (15%), followed by triplet loss (12%), contrastive loss(11%), euclidian loss (9%), and others.

2.12 Classification

The last step of a gait recognition identification process is classification. A classification layer matches the labels and classifies/arranges them to the highest predicted labels. Figure 15 depicts the most adopted classification techniques. The ratio of the most used classification technique is softmax (45%), followed by the nearest neighbor (8%), then SVM (7%), and then the rest.

Classification can be categorized as binary classification with just two classes (0 or 1) or a multi-class grouping with several levels. The process of categorizing the elements of a collection into two classes depending on a classification criterion is known as binary classification (Makihara et al. 2018). For binary classification problems, logistic regression is the go-to approach (Tong et al. 2019). A sigmoid function is a mathematical equation with a distinctive "S"-shaped form, also known as a sigmoid curve, and described by the formula S(x) = 1/1+e-x (Liu and Liu 2020). Backpropagation is a supervised learning algorithm for artificial neural networks that employ gradient descent (Song et al. 2019). Regression is a set of mathematical procedures for measuring the associations between a dependent variable (often called the 'outcome variable') and one or more independent variables (often referred to as predictors, covariates, or features). Linear regression is the most general form of regression analysis, in which a researcher seeks the line that best matches the data according to a certain mathematical criterion (Das et al. 2019). The softmax classifier takes its name from the softmax algorithm, which is used to squash raw class scores into normalized positive values that amount to one, allowing the implementation of cross-entropy loss (Shiraga et al. 2016; Sakai et al. 2019; Zhu et al. 2018).

A classification layer implements a technique for determining cross-entropy loss in multi-class classification problems involving mutually exclusive groups. SVM (supervised machine learning) algorithms are supervised learning algorithms that may be used to solve problems like classification and regression. The data is modified using a technique known as the kernel trick, and then an ideal border between the different transformation outputs is determined using this method (Chen et al. 2018). Linear SVM seeks the hyperplane with a gap between the different data points in the training set so that it classifies data with linear line (Yeoh et al. 2016). Multi SVMs are typically built by merging many two-class SVMs (Mehmood et al. 2020). The nearest centroid classifier, also known as the prototype classifier, is a classification model that labels observations based on the mean (centroid) of the training sample class, which is closest to the observation (Alotaibi and Mahmood 2017). The nearest neighbor analysis quantifies the propagation or distribution of something through a spatial space. It gives a numerical meaning representing how clustered or uniformly distributed a group of points can be (Sokolova and Konushin 2017). The K nearest neighbors approach gathers all available samples and categorizes them into similarity matrix (e.g., distance functions) (Mehmood et al. 2020). The fully connected layer uses the outputs of the convolution/pooling stage to classify them to the predicted output (Zhang et al. 2019f).

Cross reconstruction attempts to create latent relations between various modalities, thus reducing feature space distribution variations (Li et al. 2020a). The discriminator also acts as a classifier. It attempts to differentiate between actual data and data produced by the generator. It could employ any network design suitable to the form of data being classified (Zhang et al. 2019e). Wasserstein generative adversarial network is a GAN extension that increases model consistency during learning and offers a loss function that corresponds with image quality (Uddin et al. 2019a). The autoencoder classification method is like anomaly detection, which learns the pattern of a regular operation (Zhang et al. 2019a). Latent Semantic Analysis is used for extracting and expressing the contextual use-value of terms using mathematical computations on a vast corpus of text (Ling 2019). A directed acyclic graph framework in which each node contains a classifier. Classification algorithms are designed to classify class labels that are located near the root of the hierarchy (Castro et al. 2020). In the context of a tree structure, a decision tree constructs classification or regression models. It incrementally divides data into smaller subsets while developing an associated decision tree. Consequently, a tree containing decision and leaf nodes is created (Jia et al. 2017). Triplet loss is often seen in situations where the number of groups is uncertain, such as gait identification. With a triplet loss, qualified embedding can quickly determine if two gaits are similar and can set a threshold to determine if they belong to the same individual or not (Zhang et al. 2019c). The Metric learning approach is used for automatically building distance measures from supervised results (Ben et al. 2019). The horizontal pyramid pooling method is resistant to object deformations (Chao et al. 2019).

When using triplet training pairs, the loss function is always triplet ranking loss (Xu et al. 2020). Recurrent attention models are neural networks that enable the network to concentrate on aspects of a complicated input one at a time before the whole dataset is classified (Haque et al. 2016). On a multilayer feed-forward neural network, the backpropagation algorithm conducts the learning. It learns a set of weights for repeatedly predicting tuples' class labels. Uddin et al. (2019b). LSTM is a recurrent neural network that does classification based on time series data (Battistone and Petrosino 2019). The length of a line segment connecting two points in euclidean space is described as the euclidean distance (Fan et al. 2020). The histogram distance is a vector and probabilistic metric. A histogram is treated as a fixed-dimensional parameter in the vector method. As a result, regular vector norms, including city block, euclidean, and intersection, can be used as distance scales (Jia et al. 2019). GaitSet extracts and classifies all spatial and temporal data as a reference or series method more accurately and reliably than current gait (He et al. 2020). A linear layer learns the average coefficient of correlation between the output and the input with no bias; for example, w will be positive if x and y are positively linked, and w will be negative if x and y are negatively correlated (Xue et al. 2020). An All-layer margin increases the chances of misclassification in deep networks. The relationship between normalized output margin and generalization is captured in a plain and easy bound for linear classifiers where a broad output margin implies strong generalization (Xu et al. 2019).

2.13 System configuration, GPU details, framework and language used

A graphics processing unit (GPU) is a small computer dedicated to a particular task. It differs from a CPU that does several jobs at the same time. GPUs have their processors,

Table 4 System configuration, GPU detail, Language used and Framework details

Cite	System configuration	Platform (CPU/ GPU)	Language	Framework
Babaee et al. (2018a)	Ubuntu 14.04 LTS 64-bit, with one NVIDIA GeForce Titan X graphics card having 12GB of memory	NVIDIA GeForce Titan X		
Babaee et al. (2019a)	machine with a memory of 16GB and a GeForce GTX 1060 6GB GPU	GTX 1060 6GB GPU.		
Xia et al. (2019)	machine with a NVIDIA TitanX GPU	NVIDIA TitanX GPU		
Das et al. (2019)	96 GB RAM, one i9-18 core processor, along with three GPUs: one Titan Xp with 12 GB RAM, 12 GB frame-buffer memory and 256 MB BAR1 memory and, two GeForce GTX 1080 Ti with 11 GB RAM, 11 GB frame-buffer memory and 256 MB of BAR1 memory	Titan Xp with 12 GB RAM, GeForce GTX 1080 Ti with 11 GB RAM, 12 GB frame-buffer memory and 256 MB BAR1 memory		
Yeoh et al. (2016)	System with NVIDIA GTX 970 4GB GPU	NVIDIA GTX 970 4GB GPU	Matlab	CUDA and cuDNN
Castro et al. (2017a)	32 cores at 2 GHz, 256 GB of RAM and a GPU Nvidia Tesla K40c	GPU Nvidia Tesla K40c	Matlab 2014b	MatConvNet, CUDA and cuDNN
Yeoh et al. (2017)	CPU	CPU		Caffe framework
Sun and Liu (2018)	Single NVIDIA GeForce GTX 1080 GPU with 8GB on board memory	NVIDIA GeForce GTX 1080 GPU		TensorFlow
Yang et al. (2019)	computer with 8 cores at 2.1 GHz, 32 GB of RAM and a GPU Nvidia GeForce GTX 1080 Ti	GPU Nvidia GeForce GTX 1080 Ti		
Li et al. (2019b)	System with NVIDIA GeForce GTX TITAN X GPU with 12 G memory	NVIDIA GeForce GTX TITAN X GPU		Caffe
Castro et al. (2019)	Intel Xeon E5-2620 server and four PCI 3.0 slots to hold up to two Nvidia Titan X Pascal and two Titan X Maxwell GPUs	Nvidia Titan X Pascal	Python; matlab	Caffe TensorFlow, CNTK, MatConvNet and PyTorch; cuDNN

Table 4 (continued)

Cite	System configuration	Platform (CPU/ GPU)	Language	Framework
Luo and Tjahjadi (2020)	PC with an Intel Core i7(3.6GHz) CPU and 8GB RAM	CPU		
Yan et al. (2015)	Core i7-3770 3.40-GHz computer. NVIDIA GTX Titan GPU	NVIDIA GTX Titan GPU	C++	GPU CNN package in C++ language based on NVIDIA CUDA and cuDNNv2
Yu et al. (2017)	CPU			Caffe software
Zhang et al. (2019a)	Desktop with GeForce GTX 1080 Ti GPU	GeForce GTX 1080 T		
Hawas et al. (2019b)	desktop with GeForce GTX 1080 Ti GPU	GTX 1080 Ti GPU	Python	PyTorch Framework
Castro et al. (2020)	Ubuntu 16.04. 32 cores at 2.3 GHz, 256 GB of RAM and a GPU Nvidia Titan X Pascal	Nvidia Titan X Pascal	Matlab 2016a	MexConv3D MatConvNet library
Mehmood et al. (2020)	Core i7 machine containing 8 GB RAM and 4 GB NVIDIA GeForce 940MX GPU	4 GB NVIDIA GeForce 940MX GPU	MATLAB 2018b	Matconvnet
Zhang et al. (2019b)	CPU			Tensorflow
Chen et al. (2017)	System with NVIDIA GeForce GTX Titan X	GTX Titan X		Torch 7
Luo et al. (2019)	System with NVIDIA TITAN V GPUs	TITAN V		PyTorch
Ben et al. (2019)	desktop with Intel(R) Core (TM) i5-6300U CPU@2.40Hz and 8GB RAM	cpu	Matlab	
Zhang et al. (2019d)	Desktop with 8 Nvidia Titan V GPUs	8 Nvidia Titan V GPUs		
Chao et al. (2019)	PyTorch 8 NVIDIA 1080TI GPU	8 NVIDIA 1080TI GPU	Python 3.6	PyTorch 0.4 , nstall CUDA 9.0, cuDNN7.0
Zhang et al. (2019f)	System with GPU Titan X cards for	GPU Titan X cards for		

Table 4 (continued)

Cite	System configuration	Platform (CPU/ GPU)	Language	Framework
Khan et al. (2020)	System with Intel i7-6700K CPU, 64GB RAM and a NVidia GTX TITAN X GPU	NVidia GTX TITAN X GPU		keras with tensorflow
Cai et al. (2019)	System with GPUs 2 Tesla P100	GPUs 2 Tesla P100		
Castro (2017b)	PC, Ubuntu 14.04 with 32 cores at 2.2 GHz, 256 GB of RAM and a GPU NVIDIA Titan X Pascal	GPU NVIDIA Titan X Pascal,	Matlab 2016a	Caffe, MatConvNet library
Karianakis et al. (2017)	NVIDIA Tesla K80 GPU, 24 Intel Xeon E5 cores and 64G RAM memory	NVIDIA Tesla K80 GPU	Python	Caffe to Torch
Batchuluun et al. (2018)	Ubuntu with 8-GB of memory (NVidia GeForce GTX 1070) Wang et al. (2019b), and a desktop computer with intel CPU (core i7-6700 CPU @ 3.40GHz (8 CPUs)), and RAM (32,768 MB)	NVidia GeForce GTX 1070	Python	Keras application programming interface (API) with tensorflow as the backend engine
Liu et al. (2018)	CPU			caffe
Karianakis (2018)	CPU			Caffe-to-Torch
Battistone and Petrosino (2019)	Regular system	cpu	Python	TensorFlow, Scikit-learn, Numpy, OpenCV
Wu et al. (2015)	GPU is GTX 1080Ti GAMINGX with 11G memory, the CPU is AMD Ryzen 5 2600X, the main frequency is 3.6GHz, and the memory size is 32GB.	GPU is GTX 1080Ti GAMINGX with 11G memory		Keras
Liu and Liu (2020)	System with 4 Titan X (12 GB) GPU	4 Titan X (12 GB) GPU		
Rauf et al. (2016)	System with Nvidia GeForce GTX TITAN X Graphic card	Nvidia GeForce GTX TITAN X Graphic card	Python	
McLaughlin et al. (2016)	System with GTX-980 GPU	GTX-980 GPU		

Table 4 (continued)

Cite	System configuration	Platform (CPU/ GPU)	Language	Framework
Sokolova and Konushin (2017)	computer with 12 cores, 32 GB of RAM and a GPU Nvidia GTX 1070	GPU Nvidia GTX 1070		OpenCV
Nithyakani et al. (2019)	computer with 32 GB of RAM and a GPU Nvidia	GPU Nvidia		
Song et al. (2019)	system with Nvidia TitanX GPU	Nvidia TitanX GPU		
Arshad et al. (2022)	CPU		MATLAB2018a	
Delgado-Escaño et al. (2020)	Two Xeon E5-2698 16 core processors, 256 GB of RAM and a NVidia Titan X. Ubuntu 18.04	NVidia Titan X.		Keras and Tensorflow
Bhanu and Govindaraju (2011)	PC with 2 NVIDIA 1080TI GPUs	2 NVIDIA 1080TI GPUs		
Hayfron-Acquah et al. (2003)	Intel(R) Core (TM) i7-7700K central processing unit (CPU), 8.00 GB random-access memory (RAM), and NVIDIA GeForce 1050-Ti	NVIDIA GeForce 1050-Ti		
Jia et al. (2019)	System with 1080Ti GPU	1080Ti GPU		
Wang et al. (2020b)	system with NVIDIA 1080 Ti GPU	NVIDIA 1080 Ti GPU		
He et al. (2020)	CPU			
Xue et al. (2020)	NVIDIA DGX-1 AI Supercomputer	NVIDIA DGX-1		
Zheng et al. (2016)	CPU			CaffeNet
Zhu et al. (2018)	i7 CPU at 3.2 GHz and one NVIDIA TITIAN X GPU	NVIDIA TITIAN X GPU		
Alharthi et al. (2019)	System with a NVIDIA GeForce GTX 1070 GPU	NVIDIA GeForce GTX 1070 GPU		

motherboards, vRAM (visual RAM) and a proper thermal design for ventilation and cooling. Table 4 provides an overview of platforms used by authors to run their gait recognition algorithms.

2.13.1 Platform details

GPUs are employed to run gait recognition algorithms in 35% of the cited articles, whereas CPUs are used in 65 percent.

2.13.2 System configuration

Most covariates like clothing, carrying variations, and spatial-temporal features require high system configuration with GPU, as depicted by Table 4.

2.13.3 Framework used

The most common ones are cuda and cuDNN, MatConvNet, Caffe, TensorFlow, PyTorch, and Keras.

2.13.4 Language used

The reviewed articles have used three languages- python is mostly used, followed by Matlab, and very few used C++.

3 Features of deep learning

The trend in different pattern recognition disciplines is a push towards deep learning algorithms that avoid precisely handcrafting feature extraction methods by discovering the discriminating regularities in raw data. Early gait recognition system gets less accurate results than using deep learning approaches. We provide the benefits and limitations of the deep learning pipelines in the articles reviewed with respect to covariate conditions (mentioned in Parashar et al. (2022)) in Table 5. The table shows how such models have been effectively used to recognize or identify human gait.

3.1 Shortcomings and solutions

We met several unanswered questions when analyzing a huge number of research publications and applying some of the recommended answers. The correct answers to these questions may lead to a significant expansion of knowledge and understanding in this deep learning field. For every research gap, we propose possible solutions.

3.1.1 Research gaps

Appearance-based approaches are less complicated, but their accuracy is not very precise. Using deep learning to handle covariates is still problematic due to the lower accuracy of

Table 5 Benifits and drawbacks of each listed paper along with shortcomings and solutions of DL pipeline

Ref no.	Advantage	Disadvantage
Babaee et al. (2018a)	Better speed than regular CNN	High computational cost
Babaee et al. (2018b)	High accuracy	High computational cost
Babaee et al. (2019a)	Capturing Spatial features efficiently	High temporal feature loss
Xia et al. (2019)	Capturing Spatial features efficiently	Did not capture temporal features
Babaee et al. (2019b)	Capturing Spatial features efficiently	Did not capture temporal features
Li et al. (2019a)	Optimal for pose features	Low accuracy
Das et al. (2019)	Learn discriminative information	Did not capture temporal features
He et al. (2019)	Capturing spatial features efficiently	Low accuracy
Uddin et al. (2019a)	Better results	Not consider other covariates than occlusion
Wang et al. (2019a)	Reconstruct for better recognition in future	Accuracy is low
Yeoh et al. (2016)	Efficiently capturing partial gait cycle	Accuracy is low at few instances
Alotaibi and Mahmood (2017)	High accuracy	Did not capture temporal features
Castro et al. (2017a)	No preprocessing required	Gait cycle detection is not done
Tong et al. (2017)	Improved results due to fusion of MSE	High computation cost
Yeoh et al. (2017)	Good accuracy	Less analysis on various partial gait cycles level accuracy
Sun and Liu (2018)	Robust against carrying along with occlusion	Not suitable for real time application
Yang et al. (2019)	Robust against view along with occlusion	Not generating PEI due to high dimensions of GAN
Li et al. (2019b)	Do not require gait cycle information	Not handled multiple view angles
Ling (2019)	Cloth invariant Deep model	Low accuracy
Castro et al. (2019)	Efficient on small datasets	Low accuracy on OU-ISIR dataset
Li et al. (2020a)	Works with very low-resolution images	Not good with cross gait and temporal features
Luo and Tjahjadi (2020)	Considered cross view along with clothing variation	Low accuracy
Li et al. (2020b)	Achieved optimal results	View angles have not been considered
Yan et al. (2015)	Scalable architecture	High computational complexity
Yu et al. (2017)	Improved performance	Did not capture temporal features
Yu et al. (2019)	Robust and real-time surveillance application	High computational complexity
Alotaibi and Mahmood (2017)	Multi-task learning model	View variant and high computational complexity
Yu et al. (2017)	Few training parameters are required for training network	Did not consider occlusion, viewing angles and clothing, low accuracy
Liao et al. (2017)	Considered clothing and view conditions	View variant

Table 5 (continued)

Ref no.	Advantage	Disadvantage
Yao et al. (2018)	Optical flow images to build unique features	Did not consider view angle variation
Cheheb et al. (2018)	Preprocessing not required	Computationally costly
Zhang et al. (2019a)	Hybrid selection for reducing features	Computational cost and time consuming
Thapar et al. (2019)	Good accuracy	Time-consuming
Hawas et al. (2019a)	Disentangle representation learning	Classifier is less accurate
Hawas et al. (2019b)	Obtain optimal results after tuning of DL model	Did not consider different view angles
Castro et al. (2020)	Reconstruct the loss part	Classifier is less accurate
Mehmood et al. (2020)	Removed carry object selectively	Did not consider different view angles.
Swee et al. (2014)	View angle is not required	Computationally costly
Shiraga et al. (2016)	Increased their inter-class variation	Dependency on view angle is high
Wolf et al. (2016)	View invariant	Low accuracy
Zhang et al. (2017)	Carrying and clothing invariant	Computationally costly
Li et al. (2017a)	Efficient use of auto-encoders	Low accuracy
Jia et al. (2017)	Disentangles the features of pose and appearance	Did not work on all view angles
Thapar et al. (2018)	Better performance	Few temporal features are collected
Zhang et al. (2019b)	Fusion of pose and appearance-based features	Carrying condition accuracy is low
Zhang et al. (2019c)	Efficient view transformation	Did not work on all view angles
Deng et al. (2020)	View invariant	Translation issues
Wang et al. (2019b)	Efficient feature extracting model	High computational cost
Wu et al. (2016)	Good accuracy	Not consider cross-variation, cross-view, carrying and clothing
Chen et al. (2017)	Performs well under view variation	Not considering any other covariate
Huang et al. (2018)	CNN gains discriminative features	Not considering any other covariate other than view angle
Hu et al. (2048)	Efficiently capturing spatial information	Time-consuming as requires data labelling
Luo et al. (2019)	View invariant	Not consider carry and clothing
Linda et al. (2020)	High performance, view invariant	Not consider carry and clothing
Ben et al. (2019)	Works in real-time	Not consider carry and clothing
Tong et al. (2019)	Efficient view transform model	Low accuracy on cross view

Table 5 (continued)

Ref no.	Advantage	Disadvantage
Zhang et al. (2019d)	Evaluated on inter and intraclass differences	Less accurate results
Wang et al. (2020a)	Fusion of DL models	Not considered any covariate condition
Chao et al. (2019)	Spatial heterogeneous	Not addressed carry and clothing
Zhang et al. (2019d)	View-invariant	Low accuracy
Takemura et al. (2017)	Works in real-time	Not addressed carry and clothing
Zhang et al. (2019e)	Protects temporal and spatial data	Not considered occlusion
Zhang et al. (2019f)	View-invariant	Overfitting
Wang and Yan (2020)	Works in real-time	Not addressed carry and clothing
Khan et al. (2020)	Efficient loss function	Not good with large datasets
Xu et al. (2020)	Efficient construction of GEI	Not works well in real-time
Cai et al. (2019)	Works in real-time	Low performance in carry and clothing
Feng et al. (2016)	High performance	Illumination variant
Haque et al. (2016)	Spatial features are efficiently captured	Low accuracy due to loss function
Uddin et al. (2019b)	Better performance	Not consider all view angles
Li et al. (2017b)	Takes less training time	Affected by noise and illumination
Castro (2017b)	Robustly capturing the spatiotemporal feature	Not works well in real-time
Costilla-Reyes et al. (2018)	Detect gait directly from a video	High computational complexity
Karianakis et al. (2017)	Spatial features are efficiently captured	Reduced intrasubject variations
Batchuluun et al. (2018)	Prevent overfitting	Not considered all the viewing angles
Liu et al. (2018)	Robust view transformation	Not efficiently handle posture and clothing variations
Tong et al. (2018)	Capture spatial and temporal features	Lose little information while down sampling
Karianakis (2018)	High accuracy in the absence of covariates	Did not focus on real-time and covariate conditions
Battistone and Petrosino (2019)	Capturing Spatial features efficiently	Did not capture temporal features
Wu et al. (2020)	3D CNN to capture temporal features	Computational complexity is high
Huynh-The et al. (2020)	Robust gait identification	Expensive and not feasible everywhere
Wu et al. (2015)	Capturing both spatial and temporal features	Suffers from intraclass variations and occlusion

Table 5 (continued)

Ref no.	Advantage	Disadvantage
Fan et al. (2020)	Capturing both spatial and temporal features	Not considering other covariates conditions
Liu and Liu (2020)	Capturing both spatial and temporal features	Difficulty in handling posture
Rauf et al. (2016)	Capturing both spatial and temporal features	Not works well in real-time
McLaughlin et al. (2016)	Effective with less data to train	Not good accuracy
Sokolova and Konushin (2017)	Efficient in feature extraction	Not perform well on small datasets
Marín-Jiménez et al. (2017)	Less computation time	Recognition rate is very low
Nithyakani et al. (2019)	More inter class differentiation	High computational complexity
Carley et al. (2019)	Reduces the number of parameters and storage overhead	Did not capture temporal features
Song et al. (2019)	Capturing both spatial and temporal features	Not works well in real-time
Arshad et al. (2022)	Robust in a complicated environment	High computational cost
Delgado-Escaño et al. (2020)	Supports multi-gait, multi-people	Not work on occlusion
Jia et al. (2019)	Prevented unauthorized gait	Anonymized gait looks less natural
Wang et al. (2020b)	Capturing both spatial and temporal features	Model has very low accuracy
He et al. (2020)	Invariant to changes in the view angles	Model has low accuracy
Xue et al. (2020)	Not much affected due to covariates mainly viewing angles	Lower recognition rate due to eliminating all the color information
Xu et al. (2019)	Efficient in extracting pose features	Costly as it requires high-resolution cameras
Zheng et al. (2016)	Model performs well in and clothing object variations	Does not perform well on small dataset
Makihara et al. (2018)	View invariant	Did not consider the cross-view conditions
Sakai et al. (2019)	Effective for covariates like clothes, time, and carrying objects	Did not consider the cross-view conditions
Zhu et al. (2018)	Capturing both spatial and temporal features	High computational cost
Alharthi et al. (2019)	Track multiple persons at a time	Not validating results on standard dataset

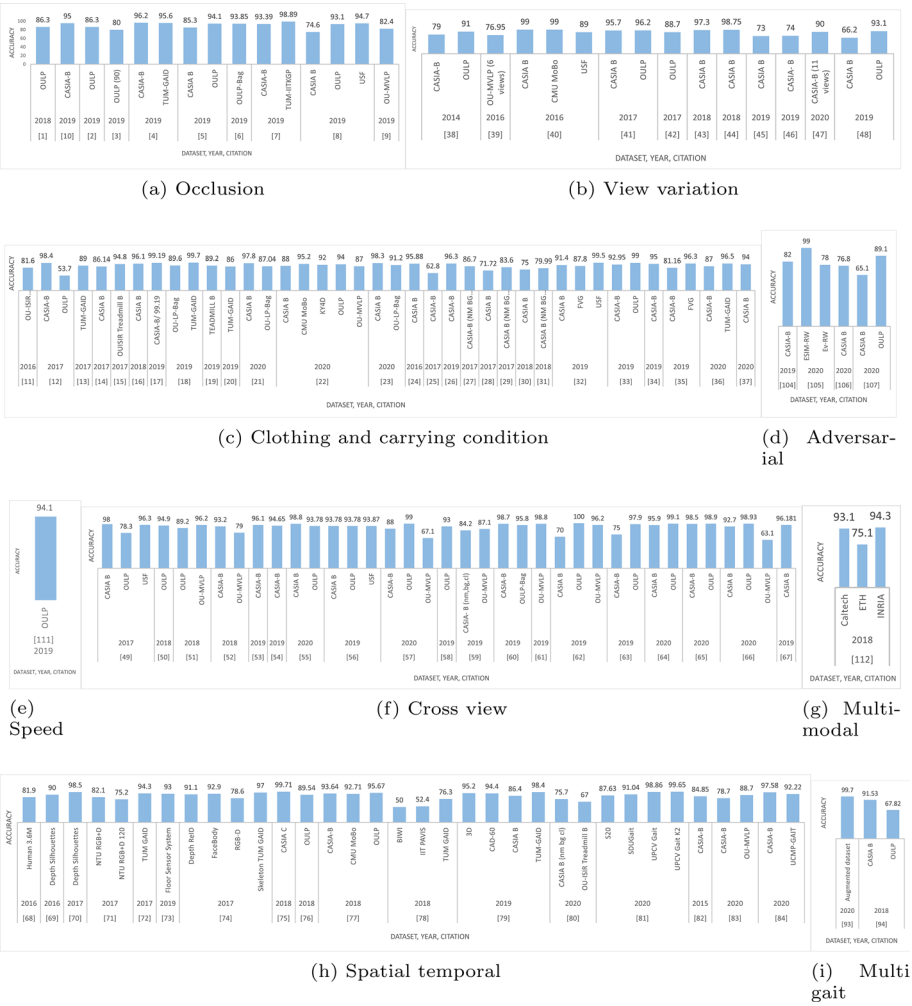


Fig. 16 Description of accuracy achieved in various covariate conditions with the dataset used

the gait recognition algorithm. The accuracy of gait detection algorithms is limited because no gait algorithm can handle real-time data. Real-time analytics is problematic because of a lack of real-time processing of deep learning architectures and massive datasets covering all relevant situations. Temporal characteristics, multi-gait, and speed variation got relatively less attention in current research. Privacy is still challenging in gait recognition systems, and there is only a tiny amount of current research devoted to security problems. 3-D datasets are more accurate than 2-D datasets, and their availability is sparse.

3.1.2 Possible solutions that can be tried

To develop an optimized deep learning model, all the parameters must be hyper tuned for as many covariate scenarios as possible. More real-time datasets should be created to

ensure that training is rigorous and to improve the recognition accuracy. Real-time analytics may be performed utilizing powerful hardware designs and by creating optimized deep architectures that operate on massive datasets and account for all covariates. To capture all the covariate situations in real-time, several gait pipelines should be created and combined with working efficiently on various datasets. For training gait recognition algorithms, hyper-tuning should be done according to real-time datasets rather than manually preprocessing their datasets. LSTM may be combined with other encoders as a deep learning pipeline to capture temporal information and speed variation. Multi-gait issues may be addressed using real-time datasets. To make it a cost-effective solution, further approaches for capturing 3D datasets or converting 2D datasets to 3D datasets should be developed.

Deep learning can automatically learn features without the need for pre-defined feature extractors. However, deep learning approaches have considerable drawbacks. Deep learning depends entirely on the dataset used to learn feature representation; as a result, the ability to generalize results may be hindered. Any covariate change in the dataset affects the output quality of the deep learning model. More real-time datasets are required to train strong deep networks with authoritative, extensive, and dependable benchmarks. As stated earlier, deep learning techniques are inherently black boxes. Through output visualization of layers, efforts have been made to comprehend the learning process of a particular network. Figure 16 gives a detailed description of the accuracy of the dataset used in each paper. Various accuracies of covariate conditions are depicted, including occlusion, view variation, clothing, carrying condition, adversarial, speed variation, cross-view, multimodal, spatial-temporal, and multi-gait.

A considerable amount of work has been done in covariate conditions like clothing, cross-view, and spatial-temporal parts while recognizing gait. On the other hand, very little work is done under multi-gait, adversarial, multimodal, anonymization, and speed variation. Using dataset TUM-IITKGP, occlusion gives the best accuracy (Fig. 16 a); however, the accuracy of the OULP, OU-MVLP and CASIA B datasets is low. Researchers should consider datasets like OULP and OU-MVLP when developing deep learning solutions. The deep learning architecture should be built in such a way that it is able to deal with any dataset. Figure 12b shows the view variation results. CASIA dataset B obtains acceptable results in this figure, but not OU-MVLP, because OU-MVLP has more subjects than CASIA B. In the OU-MVLP dataset, there are 10307 subjects, whereas, in the CASIA B dataset, there are only 124; therefore, other deep learning techniques should be implemented to improve results when using the OU-MVLP dataset. The best accuracy for clothing and carrying variations can be seen in Fig. 12 (c) using the datasets CASIA B and USF. TUM-GAID and FVG both require a significant amount of work. Only a few datasets, such as TUM-IITKGP, have not been considered in the previous research. One sequence in TUM-IITKGP is of a long gown, which can be an excellent example of clothing variation. It is important to consider the TUM-IITKGP dataset because if there is a lot of variation in a dataset to train a deep learning algorithm, then the model is capable of real-time recognition.

The true power of deep learning can be seen when training with larger datasets, such as OU-MVLP. After training, hyper tuning of the parameters must be done to improve accuracy. Figure 12 (i) (d) shows that randomly generated datasets can produce acceptable results for multi-gait and adversarial attacks. In Fig. 12f, the cross-view covariate is shown. This figure shows the best results on the OULP and CASIA B datasets. The OU-MVLP dataset (Xu et al. 2020) shows unacceptable results. In Fig. 12h, spatial and temporal covariates are given. This figure shows how well datasets like UPCV gait K2 and UPCV gait perform under spatial and temporal constraints. Datasets like BIWI and IIT

PAVIS have not produced satisfactory results, and much work in deep learning is needed to improve these datasets. Figure 12e and g show that very little work has been done in speed variation and multimodal systems, and acceptable results have not been obtained.

The use of deep learning on small datasets presents a challenge because, in general, the "power" of deep learning in recognizing patterns is proportional to the size of the dataset; the smaller the dataset, the less powerful and accurate the deep learning algorithms are. This presents a challenge for those who wish to use deep learning on small datasets because of this. Researchers have the option of using data augmentation approaches to bridge this gap.

3.2 Future direction

The deep learning frameworks that are now available perform exceptionally well in various contexts to deal with various covariate circumstances. However, their application raises a number of problems that need answering. The first challenge is making an informed decision on an appropriate framework. Next, what kinds of information can deep learning models capture? Nonetheless, how can hostile attempts readily trick the gait recognition system? In the end, what is the most basic configuration of a deep neural network capable of achieving a specific level of accuracy on a specific dataset, and how can the most successful aspects of several designs be combined to improve the system?

4 Conclusion

Let's compare the gait recognition system to other forms of biometrics, such as fingerprint, face, voice, and iris recognition. Gait recognition is still in its infancy, but its unobtrusive nature makes it more appealing than other methods for several applications. However, the covariates have hindered the effective use of this biometric in all situations. This paper identified numerous parameters of the deep learning pipeline for gait recognition covering all the covariate conditions like occlusion, clothing, carrying condition, view variation, cross-view, spatial-temporal, multi-gait, adversarial, multimodal, anonymization, and walk speed variation. Along with the accuracy of the methods, we also assessed a collection of datasets appropriate for training deep learning approaches to handle a variety of covariates. On top of that, we reviewed the most recent deep learning models for every parameter and highlighted the ones that performed exceptionally well on various covariate conditions. The possible benefits and limitations of deep learning techniques have also been listed. Research gaps were highlighted with respect to covariate-based papers using deep learning approaches. Their solutions are also proposed, and finally, we looked at the most depicted gait recognition methods of deep learning. Comparison of accuracy achieved and the datasets used are also discussed to uncover the overlooked areas in gait recognition so that they can be implemented in time to come.

Most of the literature employs automated CNN layers to collect and categorize the feature set. Hyper tuning of CNN parameters is challenging and varies from dataset to dataset. Pre-trained deep networks are widely utilized in gait surveillance systems. Still, the results are low for a few real-time datasets, while the accuracy is suitable for some manually preprocessed datasets. Parameters and layers of deep learning must be studied properly with respect to gait recognition, and values of setting parameters should be compared with the previous state-of-the-art methods.

Table 6 Feature extraction and representation details

Abb.	Full form	Abb.	Full form
DL	Deep Learning	DCNN	Deep Convolutional Neural Network
BCE	Binary Cross Entropy	DNN	Deep Neural Network
CCE	Categorical Cross Entropy	PReLU	Parametric Rectification
SGD	Stochastic Gradient Decent	ReLU	Rectified Linear Unit
MSE	Mean Square Error	LReLU	Leaky Rectified Linear Unit
WD	Weight Decay	LSTM	Long Short-Term Memory
LR	Learning Rate	FCL	Focal Convolution Layer
M	Momentum	LDA	Linear Discriminant Analysis
BP	Back Propagation	PCA	Principal Component Analysis
SNE	Stochastic Neighbor Embedding	DCT	Discrete Cosine Transform
SD	Standard Deviation	GAN	Generative Adversarial Network
MSE	Mean Square Error	BS	Background Subtraction
ED	Euclidean Distance	GEI	Gait Energy Image
LRL	Logistic Regression Loss	OF	Optical Flow
PSO	Particle Swarm Optimization	SSA	Stacked Sparse Autoencoder
RMSP	Root Mean Square Propagation	SSM	Silhouette Stereo Map
GA	Genetic Algorithm	CFA	Canonical Feature Aggregation
CGI	Chrono Gait Image	PFA	Pose Feature Aggregation
IMU	Inertial Measurement Unit	GMM	Gaussian Mixture Model
BN	Batch Normalization F	CNN	Fully Convolutional Neural Network
LRN	Local Response Normalization	HMM	Hidden Markov Model
NAC	Normalized Auto Correlation	NDNN	Non-Linear Deep Neural Network
SNR	Spectral Norm Regularization	COG	Centre of Gravity
NN	Nearest Neighbor	RNN	Recurrent Neural Network
KNN	K Nearest Neighbor	DRN	Deep Recurrent Network
NC	Nearest Centroid	MB	Model Based
SVM	Support Vector Machine	PB	Pose Based
SNN	Spiking Neural Networks	MF	Model Free
CNN	Convolution Neural Network	SNN	Siamese Neural Network
MLP	Multilayer Perceptron	BCC	Binary Cross Entropy
DCRNN	Deep Convolutional and Recurrent Neural Network	ABGAN	Alpha Blending Generative Adversarial Networks

Appendix A: Abbreviations

Abbreviations used in this paper are given in Fig. 16 in tabular form (Table 6)

Author contributions AP, AP: Conceptualization, Data acquisition, Analysis and interpretation of data, Investigation, Visualization, Writing - original draft. WD: Research Co-Supervisor, Contributed to refining the ideas, Reviewing, Editing and Finalizing this paper. RSS: Research Supervisor, Reviewing. IR: Contributed to refining the ideas, Reviewing and Finalizing this paper.

Declarations

Conflict of interest The authors declare no competing interests.

References

Alharthi AS et al (2019) Deep learning for monitoring of human gait: a review. IEEE Sens J 19(21):9575–9591

Alotaibi M, Mahmood A (2017) Reducing covariate factors of gait recognition using feature selection and dictionary-based sparse coding. Signal Image Video Process 11:1131–1138

Alotaibi M, Mahmood A (2017) Improved gait recognition based on specialized deep convolutional neural network. Comput Vis Image Underst 164:103–110

An W, Yu S, Makihara Y, Wu X, Xu C, Yu Y, Yagi Y (2020) Performance evaluation of model-based gait on multi-view very large population database with pose sequences. IEEE Trans Biom Behav Ident Sci 2(4):421–430

Arshad H, Khan MA, Sharif MI, Yasmin M, Tavares JMR, Zhang YD, Satapathy SC (2022) A multilevel paradigm for deep convolutional neural network features selection with an application to human gait recognition. Expert Syst 39(7):e12541

Babaee M, Li L, Rigoll G (2019) Person identification from partial gait cycle using fully convolutional neural networks. Neurocomputing 338:116–125

Babaee M, Li L, Rigoll G (2018) "Gait Recognition from Incomplete Gait Cycle. In: 2018 25th IEEE International conference on image processing (ICIP), pp 768–772

Babaee M, Li L, Rigoll G (2018) Gait energy image reconstruction from degraded gait cycle using deep learning. In: Proceedings of the European conference on computer vision (ECCV) workshops

Babaee M, Zhu Y, Köpüklü O, Hörmann S, Rigoll G (2019) Gait energy image restoration using generative adversarial networks. In: 2019 IEEE international conference on image processing (ICIP) (pp 2596–2600). IEEE

Barbosa IB, Cristani M, Bue AD, Bazzani L, Murino V (2012) Re-identification with rgb-d sensors. In: European conference on computer vision (pp 433–442). Springer, Berlin

Barth J, Oberndorfer C, Pasluosta C, Schülein S, Gassner H, Reinfelder S, Eskofier BM (2015) Stride segmentation during free walk movements using multi-dimensional subsequence dynamic time warping on inertial sensor data. Sensors 15(3):6419–6440

Batchuluun G et al (2018) Gait-based human identification by combining shallow convolutional neural network-stacked long short-term memory and deep convolutional neural network. IEEE Access 6:63164–63186

Battistone F, Petrosino A (2019) TGLSTM: a time based graph deep learning approach to gait recognition. Pattern Recognit Lett 126:132–138

Ben X, Gong C, Zhang P, Yan R, Wu Q, Meng W (2019) Coupled bilinear discriminant projection for cross-view gait recognition. IEEE Trans Circuits Syst Video Technol 30(3):734–747

Bhanu B, Govindaraju V (eds) (2011) Multibiometrics for human identification. Cambridge University Press, Cambridge

Cai C et al (2019) CHD: consecutive horizontal dropout for human gait feature extraction. In: International conference on computing and pattern recognition, pp 89–94

Carley C, Ristani E, Tomasi C (2019) Person re-identification from gait using an autocorrelation network. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops (pp 10–16)

Castro FM, Guil N, Marín-Jiménez MJ, Pérez-Serrano J, Ujaldón M (2019) Energy-based tuning of convolutional neural networks on multi" GPUs. Concurr Comput 31(21):e4786

Castro FM, Marín-Jiménez MJ, Guil N, de la Blanca NP (2020) Multimodal feature fusion for CNN-based gait recognition: an empirical comparison. Neural Comput Appl 32(17):14173–14193

Castro FM, Marín-Jiménez MJ, Mata NG, Blanca NP (2017) Automatic learning of gait signatures for people identification. In: LNCS, vol. 10306

Castro FM, Marín-Jiménez MJ, Guil N, López-Tapia S, de la Blanca NP (2017) Evaluation of CNN architectures for gait recognition based on optical flow maps. In: 2017 international conference of the biometrics special interest group (BIOSIG) (pp 1–5). IEEE

Chao H, He Y, Zhang J, Feng J (2019) Gaitset: regarding gait as a set for cross-view gait recognition. In: Proceedings of the AAAI conference on artificial intelligence (Vol 33, No 01, pp 8126–8133)

Cheheb I, Al-Maadeed N, Al-Madeed S, Bouridane A (2018) Investigating the use of autoencoders for gait-based person recognition. In: 2018 NASA/ESA conference on adaptive hardware and systems (AHS) (pp 148–151). IEEE
Chen X et al (2018) Multi-gait recognition based on attribute discovery. IEEE Trans Pattern Anal Mach Intell 40:1697–1710
Chen Q, Wang Y, Liu Z, Liu Q, Huang D (2017) Feature map pooling for cross-view gait recognition based on silhouette sequence images. In: 2017 IEEE international joint conference on biometrics (IJCB) (pp 54–61). IEEE
Connor P et al (2018) Biometric recognition by gait: a survey of modalities and features. Comput Vis Image Underst 167:1–27
Costilla-Reyes O, Vera-Rodriguez R, Scully P, Ozanyan KB (2018) Analysis of spatio-temporal representations for robust footstep recognition with deep residual neural networks. IEEE Trans Pattern Anal Mach Intell 41(2):285–296
Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), vol 1, pp 886–893 vol 1
Das D, Agarwal A, Chattopadhyay P, Wang L (2019) RGait-NET: an effective network for recovering missing information from occluded gait cycles
Delgado-Escaño R et al (2020) MuPeG-the multiple person gait framework. Sensors 20(5):1358
Deng M et al (2020) Human gait recognition based on deterministic learning and knowledge fusion through multiple walking views. J Franklin Inst 357(4):2471–2491
Dollar P, Wojek C, Schiele B, Perona P (2011) Pedestrian detection: an evaluation of the state of the art. IEEE Trans Pattern Anal Mach Intell 34(4):743–761
Ess A, Leibe B, Schindler K, Van Gool L (2008) A mobile vision system for robust multi-person tracking. In: 2008 IEEE conference on computer vision and pattern recognition (pp 1–8). IEEE
Fan C, Peng Y, Cao C, Liu X, Hou S, Chi J, He Z (2020) Gaitpart: temporal part-based model for gait recognition. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp 14225–14233)
Feng Y et al (2016) Learning effective Gait features using LSTM. In: ICPR, vol 0, pp 325–330
Gallego G et al (2020) Event-based vision: a survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, p 1
Giorgi G, Saracino A, Martinelli F (2021) Using recurrent neural networks for continuous authentication through gait analysis. Pattern Recognit Lett 147:157–163
Gross R, Cohn J (2001) The CMU motion of body (MoBo) dataset. Cmu-Ri-Tr-01-18(6): pp 1–11
Hadid A, Ghahramani M, Kellokumpu, V, Pietikäinen M, Bustard J, Nixon M (2012) Can gait biometrics be spoofed? In: Proceedings of the 21st international conference on pattern recognition (ICPR2012) (pp 3280–3283). IEEE
Haque A et al (2016) Recurrent attention models for depth-based person identification. In: CVPR, pp 1229–1238
Hawas AR, El-Khobby HA, Abd-Elnaby M, El-Samie A, Fathi E (2019) Gait identification by convolutional neural networks and optical flow. Multimed Tools Appl 78(18):25873–25888
Hawas AR, El-Khobby HA, Abd-Elnaby M, El-Samie A, Fathi E (2019) Gait identification by convolutional neural networks and optical flow. Multimed Tools Appl 78(18):25873–25888
Hayfron-Acquah JB, Nixon MS, Carter JN (2003) Automatic gait recognition by symmetry analysis. Pattern Recognit Lett 24(13):2175–2183
He Y, Zhang J, Shan H, Wang L (2019) Multi-task GANs for view-specific feature learning in gait recognition. IEEE Trans Inform Forensics Secur 14(1):102–113
He Z, Wang W, Dong J, Tan T (2020) Temporal sparse adversarial attack on gait recognition. arXiv preprint arXiv:2002.09674
Hirzer M, Beleznai C, Roth PM, Bischof H (2011) Person re-identification by descriptive and discriminative classification. In: Scandinavian conference on Image analysis (pp 91–102). Springer, Berlin
Hoang T, Choi D, Nguyen T (2015) On the instability of sensor orientation in gait verification on mobile phone. In: 2015 12th international joint conference on e-business and telecommunications (ICETE) (Vol 4, pp 148–159). IEEE
Horaud R, Hansard M, Evangelidis G, Ménier C (2016) An overview of depth cameras and range scanners based on time-of-flight technologies. Mach Vis Appl 27(7):1005–1020
Hu B, Guan Y, Gao Y, Long Y, Lane N, Ploetz T (2018) Robust cross-view gait recognition with evidence: a discriminant gait GAN (DiGGAN) approach. arXiv preprint arXiv:1811.10493
Huang Y, Zhang J, Zhao H, Zhang L (2018) Attention-based network for cross-view gait recognition. In: international conference on neural information processing (pp 489–498). Springer, Cham

Huynh-The T et al (2020) Learning 3D spatiotemporal gait feature by convolutional network for person identification. Neurocomputing 397:192–202
Imran J, Kumar P (2016) Human action recognition using RGB-D sensor and deep convolutional neural networks. In: 2016 international conference on advances in computing, communications and informatics (ICACCI) (pp 144–148). IEEE
Ionescu C et al (2014) Human3.6M: large scale datasets and predictive methods for 3D human sensing in natural environments. IEEE transactions on pattern analysis and machine intelligence, vol 36
Iwama H, Okumura M, Makihara Y, Yagi Y (2012) The ou-isir gait database comprising the large population dataset and performance evaluation of gait recognition. IEEE Trans Inform Forensics Secur 7(5):1511–1521
Iwashita Y et al (2010) Person Identification from Spatio-temporal 3D Gait. In: International conference on emerging security technologies, pp 30–35
Jain AK, Nandakumar K, Ross A (2016) 50 years of biometric research: accomplishments, challenges, and opportunities. Pattern Recognit Lett 79:80–105
Jia M, Yang H, Huang D, Wang Y (2019) Attacking gait recognition systems via silhouette guided GANs. In: Proceedings of the 27th ACM international conference on multimedia (pp 638–646)
Jia N, Sanchez V, Li CT (2017) Learning optimised representations for view-invariant gait recognition. In: 2017 IEEE international joint conference on biometrics (IJCB) (pp 774–780). IEEE
Karianakis N, Liu Z, Chen Y, Soatto S (2017) Person depth reid: robust person re-identification with commodity depth sensors. arXiv preprint arXiv:1705.09882
Karianakis N (2018) Reinforced temporal attention and split-rate transfer for depth-based person re-identification. In: Lecture notes in computer science, vol 11209, pp 737–756
Kastaniotis D, Theodorakopoulos I, Theoharatos C, Economou G, Fotopoulos S (2015) A framework for gait-based recognition using Kinect. Pattern Recognit Lett 68:327–335
Kastaniotis D, Theodorakopoulos I, Economou G, Fotopoulos S (2013) Gait-based gender recognition using pose information for real time applications. In: 2013 18th international conference on digital signal processing (DSP) (pp 1-6). IEEE
Khamsemanan N, Nattee C, Jianwattanapaisarn N (2017) Human identification from freestyle walks using posture-based gait feature. IEEE Trans Inform Forensics Secur 13(1):119–128
Khan MH, Farid MS, Grzegorzek M (2020) A non-linear view transformations model for cross-view gait recognition. Neurocomputing 402:100–111
Khan A, Javed MY, Alhaisoni M, Tariq U, Kadry S, Choi J, Nam Y (2022) Human gait recognition using deep learning and improved ant colony optimization. Comput Mater Cont 70:2113–2130
Kwolek B et al (2019) Calibrated and synchronized multi-view video and motion capture dataset for evaluation of gait recognition. Multimed Tools Appl 78(22):32437–32465
Li X, Mak Y et al (2019) Joint intensity transformer network for gait recognition robust against clothing and carrying status. IEEE Trans Inf Forensics Secur 14(12):3102–3115
Li X, Makihara Y, Xu C, Yagi Y, Ren M (2020) Gait recognition invariant to carried objects using alpha blending generative adversarial networks. Pattern Recognit 105:107376
Li C, Min X, Sun S, Lin W, Tang Z (2017) DeepGait: a learning deep convolutional representation for view-invariant gait recognition using joint Bayesian. Appl Sci 7(3):210
Li YR, Yu S, Wu S (2012) Pedestrian detection in depth images using framelet regularization. In: 2012 IEEE international conference on computer science and automation engineering (CSAE) (vol 2, pp 300-303). IEEE
Li X, Makihara Y, Xu C, Yagi Y, Ren M (2020) Gait recognition via semi-supervised disentangled representation learning to identity and covariate features. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp 13309–13319)
Li X, Makihara Y, Xu C, Yagi Y, Ren M (2019) Make the bag disappear: carrying status-invariant gait-based human age estimation using parallel generative adversarial networks. In: 2019 IEEE 10th international conference on biometrics theory, applications and systems (BTAS), pp 1–9
Li C, Sun S, Min X, Lin W, Nie B, Zhang X (2017) End-to-end learning of deep convolutional neural network for 3D human action recognition. In: 2017 IEEE international conference on multimedia & expo workshops (ICMEW) (pp 609–612). IEEE
Liao R, Cao C, Garcia EB, Yu S, Huang Y (2017) Pose-based temporal-spatial network (PTSN) for gait recognition with carrying and clothing variations. In: Chinese conference on biometric recognition (pp 474–483). Springer, Cham
Linda GM, Themozhi G, Bandi SR (2020) Color-mapped contour gait image for cross-view gait recognition using deep convolutional neural network. Int J Wavelets Multiresolution Inform Process 18(01):1941012

Ling H et al (2019) Attention-aware network with latent semantic analysis for clothing invariant gait recognition. Comput Mater Contin 60(3):1041–1054
Liu W et al (2018) Learning efficient spatial-temporal gait features with deep learning for human identification. Neuroinformatics 16(3–4):457–471
Liu X, Liu J (2020) Gait recognition method of underground coal mine personnel based on densely connected convolution network and stacked convolutional autoencoder. Entropy 22(6):695
Liu J et al (2019) NTU RGB+D 120: a large-scale benchmark for 3D human activity understanding. In: IEEE Transactions on Pattern Analysis and Machine Intelligence. pp 1
Luo J, Tjahjadi T (2020) View and clothing invariant gait recognition via 3d human semantic folding. IEEE Access 8:100365–100383
Luo S, Feng S, Pan H, Yin J, Zhang X (2019) A sequence-based multi-scale network for cross-view gait recognition. In: 2019 6th international conference on systems and informatics (ICSAI) (pp 1179–1183). IEEE
Makihara Y, Mannami H, Tsuji A, Hossain MA, Sugiura K, Mori A, Yagi Y (2012) The OU-ISIR gait database comprising the treadmill dataset. IPSJ Trans Comput Vis Appl 4:53–62
Makihara Y et al (2018) Gait recognition by deformable registration. In: CVPRW, vol 2018, pp 674–67410
Marín-Jiménez MJ, Castro FM, Guil N, De la Torre F, Medina-Carnicer R (2017) Deep multi-task learning for gait-based biometrics. In: 2017 IEEE international conference on image processing (ICIP) (pp 106–110). IEEE
McLaughlin N, Del Rincon JM, Miller P (2016) Recurrent convolutional network for video-based person re-identification. In: Proceedings of the IEEE conference on computer vision and pattern recognition (pp 1325–1334)
Mehmood A, Khan MA, Sharif M, Khan SA, Shaheen M, Saba T, Ashraf I (2020) Prosperous human gait recognition: an end-to-end system based on pre-trained CNN features selection. Multimed Tools Appl 5:1–21
Micucci D, Mobilio M, Napoletano P (2017) Unimib shar: a dataset for human activity recognition using acceleration data from smartphones. Appl Sci 7(10):1101
Mueggler E et al (2017) The event-camera dataset and simulator: event-based data for pose estimation, visual odometry, and SLAM. Int J Rob Res 36(2):142–149
Munaro M, Fossati A, Basso A, Menegatti E, Van Gool L (2014) One-shot person re-identification with a consumer depth camera. In: Person Re-identification (pp 161–181). Springer, London
Nambiaret et al (2019) Gait-based person re-identification: a survey. ACM Comput Surv (CSUR) 52(2):1–34
Ngo TT, Makihara Y, Nagahara H, Mukaigawa Y, Yagi Y (2014) The largest inertial sensor-based gait database and performance evaluation of gait-based personal authentication. Pattern Recognit 47(1):228–237
Nithyakani P, Shanthini A, Ponsam G (2019) Human gait recognition using deep convolutional neural network. In: 2019 3rd international conference on computing and communications technologies (ICCCT) (pp 208–211). IEEE
Özdemir A (2016) An analysis on sensor locations of the human body for wearable fall detection devices: principles and practice. Sensors 16(8):1161
Parashar A, Shekhawat RS, Ding W, Rida I (2022) Intra-class variations with deep learning-based gait analysis: a comprehensive survey of covariates and methods. Neurocomputing
Rauf M, Song C, Huang Y, Wang L, Jia N (2016) Knowledge transfer between networks and its application on gait recognition. In: 2016 ieee international conference on digital signal processing (DSP) (pp 492–496). IEEE
Roy A, Sural S, Mukherjee J, Rigoll G (2011) Occlusion detection and gait silhouette reconstruction from degraded scenes. Signal Image Video Process 5(4):415–430
Saber S, Amin KM, Adel Hammad M (2021) An efficient person re-identification method based on deep transfer learning techniques. IJCI 8(2):94–99
Sakai A, Sogi N, Fukui K (2019) Gait recognition based on constrained mutual subspace method with CNN features. In: 2019 16th international conference on machine vision applications (MVA) (pp 1-6). IEEE
Sarkar S et al (2005) The humanID gait challenge problem: data sets, performance, and analysis. IEEE Trans Pattern Anal Mach Intell 27(2):162–177
Seely RD, Samangooei S, Lee M, Carter JN, Nixon MS (2008) The university of Southampton multi-biometric tunnel and introducing a novel 3d gait dataset. In: 2008 IEEE second international conference on biometrics: theory, applications, and systems (pp 1–6). IEEE
Sepas-Moghaddam A, Etemad A (2022) Deep gait recognition: a survey. In IEEE transactions on pattern analysis and machine intelligence

Shahroudy A, Liu J, Ng TT, Wang G (2016) Ntu rgb+ d: a large scale dataset for 3d human activity analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition (pp 1010–1019)

Shiraga K, Echigo T, Yagi Y (2016) GEINet: view-invariant gait recognition using a convolutional neural network. In: International conference on biometrics, pp 1–8

Sokolova A, Konushin A (2017) Gait recognition based on convolutional neural networks. In: International archives of the photogrammetry, remote sensing & spatial information sciences, p 42

Song C, Huang Y, Huang Y, Jia N, Wang L (2019) Gaitnet: an end-to-end network for gait based human identification. Pattern Recognit 96:106988

Spagnolo P, Moeslund TB (eds.) (2014) Activity monitoring by multiple distributed sensing. Springer

Stylios I, Kokolakis S, Thanou O, Chatzis S (2021) Behavioral biometrics & continuous user authentication on mobile devices: A survey. Inform Fus 66:76–99

Sucerquia A, López JD, Vargas-Bonilla JF (2017) SisFall: a fall and movement dataset. Sensors 17(12):198

Sun Y, Liu Q (2018) Attribute recognition from clothing using a Faster R-CNN based multitask network. Int J Wavelets Multiresolution Inf Process 16(02):1840009

Sun Y, Zhang M, Sun Z, Tan T (2017) Demographic analysis from biometric data: achievements, challenges, and new frontiers. IEEE Trans Pattern Anal Mach Intell 40(2):332–351

Sung J, Ponce C, Selman B, Saxena A (2012) Unstructured human activity detection from RGBD images. In: 2012 IEEE international conference on robotics and automation (pp 842–849). IEEE

Swee W et al (2014) Gait recognition for person tracking across camera. Comput Vis ECCV Work 892:5–69

Takemura N, Makihara Y, Muramatsu D, Echigo T, Yagi Y (2017) On input/output architectures for convolutional neural network-based cross-view gait recognition. IEEE Trans Circuits Syst Video Technol 29(9):2708–2719

Takemura N, Makihara Y, Muramatsu D, Echigo T, Yagi Y (2018) Multi-view large population gait dataset and its performance evaluation for cross-view gait recognition. IPSJ Trans Comput Vis Appl 10(1):1–14

Tan D, Huang K, Yu S, Tan T (2006) Efficient night gait recognition based on template matching. In: 18th international conference on pattern recognition (ICPR'06) (Vol 3, pp 1000–1003). IEEE

Thapar D, Jaswal G, Nigam A, Arora C (2019) Gait metric learning siamese network exploiting dual of spatio-temporal 3D-CNN intra and LSTM based inter gait-cycle-segment features. Pattern Recognit Lett 125:646–653

Thapar D, Nigam A, Aggarwal D, Agarwal P (2018). VGR-net: a view invariant gait recognition network. In: 2018 IEEE 4th international conference on identity, security, and behavior analysis (ISBA) (pp 1–8). IEEE

Tieu NDT, Nguyen HH, Nguyen-Son HQ, Yamagishi J, Echizen I (2017) An approach for gait anonymization using deep learning. In: 2017 IEEE workshop on information forensics and security (WIFS) (pp 1-6). IEEE

Tong SB, Fu YZ, Ling HF (2019) Cross-view gait recognition based on a restrictive triplet network. Pattern Recognit Lett 125:212–219

Tong S, Fu Y, Yue X, Ling H (2018) Multi-view gait recognition based on a spatial-temporal deep neural network. IEEE Access 6:57583–57596

Tong S, Fu Y, Ling H (2017) Verification-based pairwise gait identification. In: 2017 IEEE international conference on multimedia & expo workshops (ICMEW) (pp 669–673). IEEE

Uddin MZ, Muramatsu D, Takemura N, Ahad MAR, Yagi Y (2019) Spatio-temporal silhouette sequence reconstruction for gait recognition against occlusion. IPSJ Trans Comput Vis Appl 11(1):1–18

Uddin M, Ngo TT, Makihara Y, Takemura N, Li X, Muramatsu D, Yagi Y (2018) The ou-isir large population gait database with real-life carried object and its performance evaluation. IPSJ Trans Comput Vis Appl 10(1):1–11

Uddin MZ, Khaksar W, Torresen J (2017) A robust gait recognition system using spatiotemporal features and deep learning. In: 2017 IEEE international conference on multisensor fusion and integration for intelligent systems (MFI) (pp 156–161). IEEE

Vemulapalli R, Arrate F, Chellappa R (2014) Human action recognition by representing 3d skeletons as points in a lie group. In: Proceedings of the IEEE conference on computer vision and pattern recognition (pp 588–595)

Wang Liang et al (2003) Silhouette analysis-based gait recognition for human identification. IEEE Trans Pattern Anal Mach Intell 25(12):1505–1518

Wang Y, Song C, Huang Y, Wang Z, Wang L (2019) Learning view invariant gait features with two-stream GAN. Neurocomputing 339:245–254

Wang X, Yan WQ (2020) Human gait recognition based on frame-by-frame gait energy images and convolutional long short-term memory. Int J Neural Syst 30(01):1950027
Wang X, Zhang J, Yan WQ (2020) Gait recognition using multichannel convolution neural networks. Neural Comput Appl 32(18):14275–14285
Wang K, Liu L, Lee Y, Ding X, Lin J (2019) Nonstandard periodic gait energy image for gait recognition and data augmentation. In: Chinese conference on pattern recognition and computer vision (PRCV) (pp 197–208). Springer, Cham
Wang L, Kim TK, Yoon KJ (2020) Eventsr: from asynchronous events to image reconstruction, restoration, and super-resolution via end-to-end adversarial learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp 8315–8325)
Wang Y et al (2016) Gait recognition based on 3D skeleton joints captured by kinect. In: ICIP, pp 3151–3155
Wang T, Gong S, Zhu X, Wang S (2014) Person re-identification by video ranking. In: European conference on computer vision (pp 688–703). Springer, Cham
Wolf T, Babaee M, Rigoll G (2016) Multi-view gait recognition using 3D convolutional neural networks. In: 2016 IEEE international conference on image processing (ICIP) (pp. 4165–4169). IEEE
Wu X et al (2015) Gait recognition based on Densenet transfer learning. Ijset Net 9(1):1–14
Wu Z, Huang Y, Wang L, Wang X, Tan T (2016) A comprehensive study on cross-view gait based human identification with deep CNNS. IEEE Trans Pattern Analy Mach Intell 39(2):209–226
Wu X, An W, Yu S, Guo W, García EB (2020) Spatial-temporal graph attention network for video-based gait recognition. In: Asian Conference on Pattern Recognition (pp 274–286). Springer, Cham
Xia LM, Wang H, Guo WT (2019) Gait recognition based on Wasserstein generating adversarial image inpainting network. J Central South Univ 26(10):2759–2770
Xu C et al (2020) Cross-view gait recognition using pairwise spatial transformer networks. IEEE Trans Circuits Syst Video Technol 8(1):15–19
Xu Z, Lu W, Zhang Q, Yeung Y, Chen X (2019) Gait recognition based on capsule network. J Vis Commun Image Represent 59:159–167
Xu C, Makihara Y, Ogi G, Li X, Yagi Y, Lu J (2017) The OU-ISIR gait database comprising the large population dataset with age and performance evaluation of age estimation. IPSJ Trans Comput Vis Appl 9(1):1–14
Xue W, Ai H, Sun T, Song C, Huang Y, Wang L (2020) Frame-GAN: increasing the frame rate of gait videos with generative adversarial networks. Neurocomputing 380:95–104
Yan C, Zhang B, Coenen F (2015) Multi-attributes gait identification by convolutional neural networks. In: 2015 8th international congress on image and signal processing (CISP) (pp 642–647). IEEE
Yang F et al (2019) Gait recognition with clothing and carrying variations based on GEI and CAPDS features. In: LNCS
Yao L, Kusakunniran W, Wu Q, Zhang J, Tang Z (2018) Robust CNN-based gait verification and identification using skeleton gait energy image. In: 2018 digital image computing: techniques and applications (DICTA) (pp 1–7). IEEE
Yeoh T, Aguirre HE, Tanaka K (2016) Clothing-invariant gait recognition using convolutional neural network. In: 2016 International symposium on intelligent signal processing and communication systems (ISPACS) (pp 1–5). IEEE
Yeoh TW, et al (2017) Stacked progressive auto-encoders for clothing-invariant gait recognition. In: LNCS, pp 151–161
Yu S, Chen H, Wang Q, Shen L, Huang Y (2017) Invariant feature extraction for gait recognition using only one uniform model. Neurocomputing 239:81–93
Yu S, Liao R, An W, Chen H, García EB, Huang Y, Poh N (2019) GaitGANv2: invariant gait feature extraction using generative adversarial networks. Pattern Recognit 87:179–189
Yu S et al (2006) A framework for evaluating the effect of view angle, clothing and carrying condition on gait recognition. In: ICPR'06, vol 4, pp 441–444
Yu S, Chen H, Garcia Reyes EB, Poh N (2017) Gaitgan: Invariant gait feature extraction using generative adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops (pp 30–37)
Zhang Y, Huang Y, Wang L, Yu S (2019) A comprehensive study on gait biometrics using a joint CNN-based method. Pattern Recognit 93:228–236
Zhang Y, Huang Y, Yu S, Wang L (2019) Cross-view gait recognition by discriminative feature learning. IEEE Trans Image Process 29:1001–1015
Zhang R, Yin D, Zhou Z, Cao Z, Meng F, Hu B (2019) Improving cross-view gait recognition with generative adversarial networks. Electr Eng Comput Sci (EECS) 3:43–47

Zhang Z, Tran L, Yin X, Atoum Y, Liu X, Wan J, Wang N (2019) Gait recognition via disentangled representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp 4710–4719)
Zhang X, Sun S, Li C, Zhao X, Hu Y (2017) Deepgait: a learning deep convolutional representation for gait recognition. In: Chinese conference on biometric recognition (pp. 447–456). Springer, Cham
Zhang P, Wu Q, Xu J (2019) VN-GAN: identity-preserved variation normalizing GAN for gait recognition. In: 2019 international joint conference on neural networks (IJCNN) (pp 1–8). IEEE
Zhang P, Wu Q, Xu J (2019) VT-GAN: View transformation GAN for gait recognition across views. In: 2019 international joint conference on neural networks (IJCNN) (pp 1–8). IEEE
Zhang K, Luo W, Ma L, Liu W, Li H (2019) Learning joint gait representation via quintuplet loss minimization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp 4700–4709)
Zheng L, Bie Z, Sun Y, Wang J, Su C, Wang S, Tian Q (2016) Mars: a video benchmark for large-scale person re-identification. In: European conference on computer vision (pp 868–884). Springer, Cham
Zheng J, Liu X, Gu X, Sun Y, Gan C, Zhang J, Yan C (2022) Gait recognition in the wild with multi-hop temporal switch. arXiv preprint arXiv:2209.00355
Zhu C et al (2018) Effective human detection via multi-model classification and adaptive late fusion. Int J Wavelets Multiresolution Inf Process 16(02):1840012

Authors and Affiliations

Anubha Parashar[1] · Apoorva Parashar[2] · Weiping Ding[3] · Rajveer S. Shekhawat[1] · Imad Rida[4]

Apoorva Parashar
apoorvaparashar0000@gmail.com

Rajveer S. Shekhawat
rajveer1957@gmail.com

Imad Rida
rida.imad@gmail.com

[1] School of Computer Science and Engineering, Manipal University Jaipur, Jaipur, Rajasthan 303007, India

[2] Emerging Technology, Mahindra Integrated Business Solutions, Mumbai, Maharashtra, India

[3] School of Information Science and Technology, Nantong University, Nantong, China

[4] BMBI Laboratory, University of Technology of Compiègne, 60200 Compiègne, France