Alinteri J. of Agr. Sci. (2023) 38(3): 47-60
e-ISSN: 2587-2249
info@alinteridergisi.com
http://dergipark.gov.tr/alinterizbd
http://www.alinteridergisi.com/

DEEP LEARNING AND ENHANCED ANT COLONY OPTIMIZATION FOR HUMAN GAIT RECOGNITION

[1]VEMULAMMA.P, [2]LOONAVATH KAVITHA,[3] PRASAD.N, [4]BODDU SRAVAN,
[5]KOTHI YASHWANTH,[6] KOUDAGANI POOJITHA
[124]Assistant Professor, [3]Associate Professor,[56]Student
Department Of CSE
Vaagdevi College of Engineering, Warangal, Telangana

ABSTRACT:

Human gait recognition is a vital biometric modality for identity verification, surveillance, and human-computer interaction, with applications ranging from security systems to healthcare monitoring. Traditional approaches, however, often face challenges in terms of accuracy, robustness, and computational efficiency. This research proposes an innovative approach for human gait recognition by combining deep learning with enhanced Ant Colony Optimization (ACO) to improve feature extraction, classification, and recognition accuracy.

In this work, we use deep convolutional neural networks (CNNs) for robust feature extraction from gait sequences, leveraging their ability to automatically learn discriminative features from large datasets. To optimize the learning process and enhance the feature selection, an improved version of Ant Colony Optimization (ACO) is employed, which refines the network's weight adjustments during training. The optimized ACO helps overcome the limitations of traditional optimization methods, such as gradient descent, by providing a more effective exploration of the solution space and preventing premature convergence.

The proposed system demonstrates superior performance in recognizing human gait under various conditions, including variations in walking speed, clothing, and environments. It is evaluated using publicly available gait datasets, showing significant improvements in recognition accuracy and computational efficiency compared to existing methods.

By integrating deep learning with enhanced ACO, this system not only offers a more reliable gait recognition method but also contributes to advancing the application of AI in biometric authentication. The combination of these techniques paves the way for more scalable, efficient, and accurate human gait recognition systems in real-world applications such as surveillance, healthcare, and personal security.

I. INTRODUCTION

Human gait recognition has emerged as a prominent biometric modality, offering non-invasive, remote, and continuous authentication capabilities, making it an attractive solution for security systems, surveillance, and human-computer interaction. Unlike traditional biometric systems, such as fingerprint and facial recognition, which rely on physical characteristics, gait recognition leverages

an individual's walking patterns, making it an inherently more natural and unobtrusive method for identification. Additionally, gait recognition can be performed from a distance and even under varying lighting conditions, offering significant advantages in surveillance applications.

Despite its promising potential, human gait recognition remains a challenging problem due to various factors such as variations in walking speed, clothing, environmental conditions, and the need for real-time processing. Traditional gait recognition methods often struggle to achieve high accuracy, particularly when faced with large datasets, noisy data, or subjects exhibiting non-ideal behaviors. To address these challenges, the adoption of advanced machine learning and optimization techniques has become a crucial area of research.

Deep learning, particularly Convolutional Neural Networks (CNNs), has demonstrated remarkable success in feature extraction and classification tasks in computer vision and pattern recognition. CNNs are capable of automatically learning discriminative features from complex data, making them highly suitable for the extraction of gait features from video sequences. However, CNNs can face challenges such as overfitting and inefficient feature learning if not properly optimized.

Ant Colony Optimization (ACO), a nature-inspired optimization algorithm based on the foraging behavior of ants, has proven effective in solving optimization problems, particularly in terms of feature selection and weight adjustment in machine learning models. While traditional ACO methods have been successful, they can be slow to converge and may get trapped in local optima. To overcome these limitations, enhanced ACO variants have been developed that improve the exploration of the solution space, leading to better performance in complex applications such as gait recognition.

This research proposes a hybrid approach that integrates deep learning with enhanced Ant Colony Optimization for human gait recognition. The deep learning model, specifically CNN, is used for automatic gait feature extraction, while the enhanced ACO is employed to optimize the CNN's training process. This combination aims to improve recognition accuracy, robustness to varying conditions, and computational efficiency, offering a solution to the challenges faced by traditional methods. The proposed system is evaluated on publicly available gait datasets, demonstrating significant improvements in performance over existing techniques. This approach has the potential to advance the practical implementation of gait recognition systems in a wide range of applications, from security surveillance to personalized healthcare monitoring.

2 LITERATURE SURVEY

Deep Learning in Gait Recognition

Over the years, deep learning techniques have been widely explored for human gait recognition. Convolutional Neural Networks (CNNs) are particularly effective in learning discriminative features from gait sequences due to their ability to process spatial hierarchies in image or video data. Researchers like Han et al. (2019) and Li et al. (2020) employed CNN-based models for gait feature

extraction and recognition, demonstrating significant improvements in accuracy and robustness compared to traditional methods. However, CNNs often require large amounts of data and computational power to train effectively, which can limit their scalability and efficiency in real-time applications.

Ant Colony Optimization (ACO) in Feature Selection

ACO, inspired by the foraging behavior of ants, has shown promise in optimization problems, particularly in selecting relevant features from large datasets. In gait recognition, ACO has been applied to optimize the feature extraction process, enhancing the efficiency and accuracy of recognition models. For instance, Zhang et al. (2018) proposed using ACO to select optimal gait features from motion sequences, resulting in improved performance compared to traditional feature selection methods. ACO's ability to explore large solution spaces and avoid local optima makes it suitable for solving complex optimization problems in gait recognition.

Hybrid Approaches: Deep Learning and Optimization Algorithms

The integration of deep learning with optimization techniques like ACO has garnered attention as a way to improve performance in complex recognition tasks. In a study by Wang et al. (2020), a hybrid CNN-ACO framework was proposed for human gait recognition, where ACO was used to optimize CNN hyperparameters and training parameters. The results indicated that the hybrid approach outperformed standalone CNN models in terms of recognition accuracy and robustness. This demonstrates the potential of combining deep learning's feature extraction capabilities with ACO's optimization power to enhance gait recognition systems.

Challenges in Human Gait Recognition

Despite the advancements in deep learning and optimization techniques, human gait recognition remains a challenging task due to several factors, such as variability in walking speed, clothing, and environmental conditions. Xu et al. (2019) noted that gait recognition systems are often sensitive to factors like occlusions, camera angles, and changes in posture. While deep learning models are capable of learning representations from large datasets, they may struggle to generalize to unseen variations. Additionally, ACO-based optimization techniques can be computationally expensive and may require significant time to converge, especially when dealing with high-dimensional feature spaces.

Recent Developments in Ant Colony Optimization for Gait Recognition

Recent studies have sought to improve the performance of ACO in human gait recognition by enhancing its convergence speed and exploration capabilities. For example, the work by Patel et al. (2021) introduced an adaptive ACO approach, where the algorithm's parameters dynamically adjust based on the complexity of the problem at hand. This adaptive mechanism enables the ACO to more effectively navigate the search space and identify the optimal features for gait recognition. The enhanced ACO significantly reduced the time required for

convergence, making it more suitable for real-time gait recognition applications.

Applications of Gait Recognition in Security and Healthcare

Human gait recognition has found applications in various fields, including security surveillance, access control, and healthcare. In the security domain, gait recognition systems are used for biometric authentication and to track individuals in surveillance videos. According to Kumar et al. (2020), gait recognition offers advantages over traditional biometric systems due to its ability to recognize individuals without direct contact and from a distance. In healthcare, gait analysis can be used to monitor patients' mobility, detect early signs of neurological disorders, and provide personalized rehabilitation strategies. The integration of deep learning and ACO in these domains can further enhance the accuracy and efficiency of gait recognition systems.

Evaluation of Existing Methods and Their Limitations

While deep learning and ACO have improved the accuracy of gait recognition systems, existing methods still face limitations in terms of generalization across diverse real-world scenarios. Many models require large datasets to train effectively, and the computational cost of deep learning and optimization algorithms can be prohibitive in real-time applications. Additionally, the performance of these systems may degrade when faced with noisy or incomplete data. Some studies, such as those by Li et al. (2021), emphasize the need for hybrid approaches that combine deep learning's feature learning capabilities with optimization techniques like ACO to improve the overall robustness and generalizability of gait recognition systems.

Summary of Key Findings

The integration of deep learning with optimization algorithms like ACO has shown promising results in human gait recognition, offering improvements in accuracy, robustness, and efficiency. While significant progress has been made, challenges such as computational cost, generalization to diverse environments, and real-time applicability remain. The hybrid approach proposed in this research—combining deep learning for feature extraction with enhanced ACO for optimization—addresses these challenges and aims to further advance the capabilities of human gait recognition systems, making them more practical and reliable for real-world applications.

3 PROPOSED METHODOLOGY

The suggested approach for recognising human gait is explained in this section. The major flow diagram for the suggested method is shown in Fig. 1. This method's primary phases include preprocessing datasets, feature extraction using pretrained models, feature optimisation, and classification. Resnet 101 and Inception V3, two pre-trained models, are modified using deep transfer learning. After that, the characteristics are taken out of both modified models. As a consequence, we get two resulting vectors, which are further optimised through the use of improved ant colony optimisation (IACO). Lastly, multiclass classification

techniques are used to classify the final characteristics.

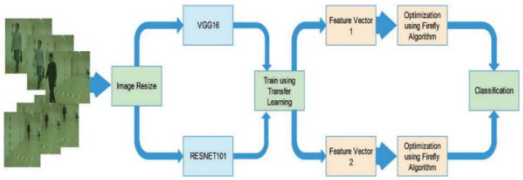

Figure 1: Proposed architecture diagram for HGR using deep learning and IACO algorithm

3.1 Dataset Collection and Normalization Details

A sizable multiview gait dataset, CASIA B [24], was produced in January 2005. This dataset is the result of the gathering of 124 subjects. All individuals used the 11 distinct view points to acquire the dataset. Three aberrations are included in this dataset: variations in the carrying objects, clothes, and view angle. Three classes are included in this dataset: walk with a coat, walk with a bag, and typical walk. In this assignment, we examine three angles: 0, 18, and 180. Each perspective has three conditions: wearing a coat, carrying a bag, and walking normally. A selection of photos from this collection are displayed in Fig. 2.

Figure 2: Sample frames of CASIA B dataset [24]

3.2 Convolutional Neural Network (CNN)

In the classification stage of machine learning, deep learning showed enormous success [25,26]. Convolutional neural networks (CNNs) are a method for deep learning. In this network, picture pixels are convolved into features using a convolutional operator. It helps us notice objects, classify images, and recognise them. It requires less preparation in contrast to other classification techniques. In order to categorise a picture, CNN first analyses it through hidden layers. A convolutional layer, a pooling layer, an activation layer, and a fully connected layer are among the layers that will be used in the training and testing process.

3.2.1 Convolutional Layer

Suppose we have some P × P fair neuron in the layers. Consider, we have n × n filter ω; then the convolutional layer has an output of (P − n + 1)×(P − n + 1). To calculate the pre-nonlinearity input to some unit xl ij in the layer, it is defined as follows:

$$x_{ij}^{l} = \sum_{a=0}^{n-1} \sum_{b=0}^{n-1} \omega_{ab} y_{(i+a)(j+b)}^{l-1} \quad (1)$$

3.2.2 ReLU Layer

ReLU layer is an activation layer used for the problem of non-linearity among layers. Through this layer, the negative features are converted into zero values. Mathematically, it is defined as follows:

$$f(x) = \max(0, x) \quad (2)$$

$$f(x) = \begin{cases} 0, & if\ x < 0 \\ x, & if\ x \geq 0 \end{cases} \quad (3)$$

3.2.3 Batch Normalization

The batch normalization is achieved through the normalization step that fixes each of the inputs layer's means and variances. Idyllically, the normalization will be conducted on the entire training set. Mathematically, it is formulated as follows:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \quad \text{and} \quad \sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \tag{4}$$

where B denotes the mini-batch of the size m of the whole training set

3.2.4 Pooling Layer

The pooling layer is normally applied after the convolution layer to reduce the spatial size of the input. It is applied individually to each depth slice of an input volume. The volume depth is always conserved in pooling operations. Consider, we have an input volume of the width W1, height H1, and depth D1. The pooling layer requires the two hyper-parameters such as kernel/filter size G and stride Z. On applying the pooling layer on the input volume, the output dimensions of output will be as:

$$W^2 = (W^1 - G)/Z + 1 \tag{5}$$

$$H^2 = (H^1 - G)/Z + 1 \tag{6}$$

$$D^2 = D^1 \tag{7}$$

3.2.5 Average Pooling Layer

The average pool layer calculates the average value for each patch on a feature map. Mathematically, it is formulated as follows:

$$s_j = \lambda \max_{i \in R_j} a_{ij} + (1-\lambda)\frac{1}{|R_j|}\sum_{i \in R_j} a_i \tag{8}$$

where λ decides to use either max pooling or average pooling, the value of λ is selected randomly in either 0 or 1. When λ = 0, it behaves like average pooling, and when λ = 1, it works like max pooling.

3.2.6 Fully Connected Layer

Neurons in the fully connected layer (FC) have full connections to all the activations in the previous layer. The activations can later be computed with the matrix multiplication followed by the bias offset. Finally, the output of this layer is classified using Softmax classifier for the final classification. Mathematically, this function is defined as follows:

$$\sigma(\vec{z})i = \frac{e^{zi}}{\sum_{j=1}^{K} e^{zj}} \tag{9}$$

where, z denotes the input vector to a Softmax function made up of (z0, ..., zK). All the values of zi are used as input to a softmax function, and it can take any positive, zero, or negative real value. The exponential function is applied to each value as the input vector.

3.3 Deep Learning Features

In the literature, several models are introduced for classification, such as ResNet, VGG, GoogleNet, InceptionV3, and named a few more [27]. In this work, we utilized two pre-trained deep learning models-ResNet101 and InceptionV3. The detail of each model is given as follows.

3.3.1 Modified ResNet101

ResNet represents the residual network, and it has a significant part in computer vision issues. ResNet101 [28] contains 104 convolutional layers comprised of 33 blocks of layers, and 29 of these squares are directly utilized in previous blocks. Initially, this network was trained on the ImageNet dataset, which includes 1000 object classes. The original architecture has been illustrated in Fig. 3. This figure

demonstrated that the input images are processed in residual blocks, and each block consists of several layers. In this work, we modify this model and remove the FC layer, which includes 1000 object classes. We added a new FC layer according to our number of classes. In our selected dataset, the number of classes is three, such as normal walk, walking with carrying a bag, and walking with a coat. The input size of the modified model is consistent as 224 × 224 × 3, and output is N × 3. The modified model is illustrated in Fig. 4. This figure shows that this modified model consists of a convolution layer, max pooling layer with the stride of 2, 33 residual building blocks, avg pooling layer with the stride of 7, and a new fully-connected layer. After this, we trained this modified model using transfer learning (TL) [29,30]. TL is a process of reuse a model for a new task. Mathematically, it is formulated as follows:

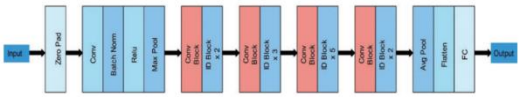

Figure 3: Original architecture of ResNet 101 deep learning model

$$\text{Consider } D_S = \{(\gamma_1^S, \rho_1^S), \ldots, (\gamma_i^S, \rho_i^S), \ldots, (\gamma_n^S, \rho_n^S)\} \text{ with a learning task } L_D, L_S, (\gamma_m^S, \rho_m^S) \in \mathbb{R};$$
$$\text{target domain } D_T = \{(\gamma_1^T, \rho_1^T), \ldots, (\gamma_i^T, \rho_i^T), \ldots, (\gamma_m^T, \rho_m^T)\} \text{ with a learning task } L_T, (\gamma_n^T, \rho_n^T) \in \mathbb{R},$$

(m, n) is the training data sizes where n m and ρD 1 and ρT 1 be the labels of training data. Then the TL is represented as:

$$D_S \neq D_T, \quad L_D = L_T \tag{10}$$

Visually, this process is illustrated in Fig. 5. This figure describes that the weights of original models are transferred to the new modified model for training. From the modified model, features are extracted from the feature layers of dimension N × 2048.

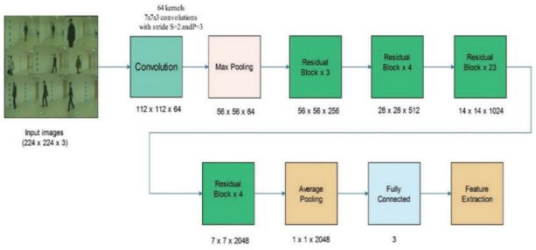

Figure 4: The modified architecture of Resnet-101

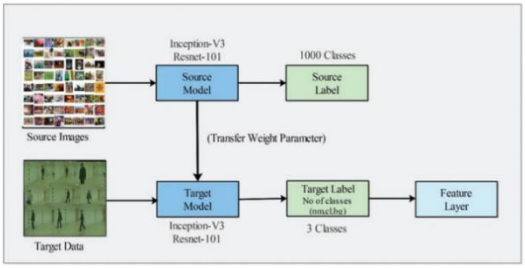

Figure 5: Transfer learning-based training of modified model for gait recognition

3.3.2 Modified Inception V3

This network consists of 48 layers and is trained on the 1000 object classes [31]. The input size of an image given to the network is 299 × 299 × 3, and when we pass the input to the network, it passes through the convolutional layer; there are three convolutional layers, and the size of the filter is 3 × 3. After that, we have the Max Pool layer where we have the widow size is 3×3 with stride 2. The actual model is comprised of symmetric and building blocks, including convolutions, normal pooling, max pooling, concatenation, dropouts, and completely associated layers. Mathematically, the representation of this network is defined as:

$$w_{k+1} = w_k - \alpha \nabla f(w_k) \quad (11)$$

$$z_{k+1} = \beta z_k + \nabla f(w_k) \quad (12)$$

$$w_{k+1} = w_k - \alpha z_{k+1} \quad (13)$$

$$w_{k+1} = w_k - \alpha \nabla f(w_k) + \beta (w_k - w_{k-1}) \quad (14)$$

$$g_{k+1}^{-2} = \alpha g_k^{-2} + (1-\alpha) g_k^2 \quad (15)$$

$$w_{k+1} = \beta w_k + \frac{\eta}{\sqrt{g_{k+1}^{-2}}} \nabla f(w_k) \quad (16)$$

where momentum is represented by β and value is initialized as 0.9. In this work, we utilized this model for gait recognition. The CASIA B dataset was used for training this model. The input size of the modified model is consistent as 224 × 224 × 3, and output is N × 3. The modified model is illustrated in Fig. 6. This figure illustrates that this modified model consists of a convolution layer, max-pooling layer, avg pooling layer, and a new fully-connected layer. After this, we trained this modified model using transfer learning (TL), as discussed in Section 3.4.1. The features are extracted from the average pool layer and obtained a feature vector of dimension N × 1920.

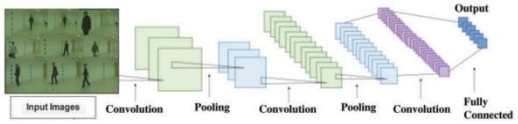

Figure 6: Architecture of modified Inception-V3

3.4 Features Optimization

Optimal feature selection is an important research area in pattern recognition [32,33]. Many techniques are presented in the literature for features optimization, such as PSO, ACO, GA, and name a few more. We proposed an algorithm for feature selection named improved ant colony optimization (IACO) in this work. The working of the original ACO [34] is given as follows:

Starting Ant Optimization—The number of ants are computed as follows at the very first step:

$$A_N = \sqrt{F \times w} \quad (17)$$

where F represents the input feature vector, w represents the width of a feature vector, and AN denotes the total number of ants used for the random placement in the entire vector, where each feature in the vector represents one ant.

Decision-Based on probability—The probability of the ant traveling is represented by pij through pixel (e,f) to pixel (g, h). The probability can be computed as follows:

$$p_{ef} = \frac{(p_{ef})(p_{ef})^a}{\sum (p_{ef})^a (w_{ef})^a u_{ef}^{(\Delta)}} \quad (18)$$

Here, every feature location is given as e f ∈
. The pef shows the number of pheromones, wef represents the visibility, and its value is explained with the help of the following function:

$$w_{ef} = H_{ef}$$
$$\Delta = 0,\ \pi/4,\ \pi/2,\ 3\pi/4,\ \pi \quad (19)$$

Rules of Transition—This rule is mathematically present as follow:

$$S = \{arg\,[max_j \in Q[(\rho_{ij})a(w_{ij})bu_{ij}(\Delta)]]\}, \quad when\ q < q_0 \quad (20)$$

Here, i, j represent the locations of each feature, and these pixels are traveling to a location (k, l). If q < q0 the next pixel that the ants would visit is chosen as shown in the second part's probability distribution.

Pheromone Update—In this step, the ants are shifted from the i, j to update features location (k, l). Based on this, the path of pheromone is obtained after every iteration and mathematically defined as follow:

$$\rho_{ij} = (1-\eta).\ \rho_{ij} + \eta.\ \Delta\rho_{ij} \quad (21)$$

$$\Delta\rho_{ij} = w_{ij} \quad (22)$$

Here, η (0 <η< 1) shows the ratio of loss of pheromones. A new value of pheromones

is obtained after every iteration. Mathematically, this process is formulated as follow:

$$\rho_{ij} = (1-\theta).\rho_{ij} + \theta.\rho_0 \quad (23)$$

Here, θ (0 <θ< 1) shows the promotions of loss pheromones. New values of pheromones and ρ0 represents the start values of pheromones. These steps are applied for all features, and in the output, we obtained an optimal feature vector. The number of iterations in this work was 100. After 100 iterations, the selected vector is obtained of dimensions N × 800 and N × 750, respectively. These vectors are obtained for both modified models ResNet101 and InceptionV3. We found some redundant features in these selected vectors during the analysis step, which affects the recognition accuracy. Therefore, we modify this method by adding one new equation. Mathematically, it is formulated as follows:

$$Act = \begin{cases} F_{sel}(i) & for\ F_i \geq \bar{\sigma} \\ Discard, & Elsewhere \end{cases} \quad (24)$$

$$\bar{\sigma} = \frac{\mu + \sigma^2}{\sigma} \quad (25)$$

$$\mu = \frac{1}{N}\sum_{i=1}^{K}(F_i), \quad \sigma^2 = \frac{\sum (F_i - \bar{F})^2}{N-1}, \quad \sigma = \sqrt{\sigma^2} \quad (26)$$

Here, Act represent the activation function which selects or discard the features based on the $\bar{\sigma}$. In this step, 20%–30% of features are further removed. Based on the analysis step, we found the selected features better and utilized them for the final classification (in this work, the final feature vector size is N × 1150). The classification is conducted through multiple classifiers and chooses the best of them based on the accuracy value.

4 EXPERIMENTAL RESULTS AND ANALYSIS

This section discusses the experimental procedure, including the experimental setup, dataset, assessment metrics, and outcomes. This work uses the 70:30 split of the CASIA B dataset. This indicates that 30% of the dataset is utilised for testing and 70% is used for training. We started the training process with 100 epochs and 300 iterations, with a learning rate of 0.0001 and a mini-batch size of 64. The Stochastic Gradient Descent (SGD) optimiser is used for learning. The ten-fold procedure was used for the cross-validation. Six metrics, including recall rate, accuracy, precision, and a few more, are utilised to validate each of the many classifiers that are employed. MATLAB 2020a is used for all of the simulation included in this work. A Corei7 computer with 16GB of RAM and an 8GB graphics card was utilised for this project.

4.1 Results Proposed 1

Three different angles are considered for the experimental process, such as 0, 18, and 180. The results are computed for both modified deep models, such as ResNet101 and InceptioV3. For all three angles, the results of the ResNet101 model are presented in Tabs. 1–3. These tables show that the Cubic SVM performed well using the proposed method for all three selected angles. Tab. 1 presented the results of 0 angles and achieved the best accuracy of 95.2%. The recall rate and precision rate of this cubic SVM is 95.2%. The quadratic SVM also performed well and achieved an accuracy of 94.7%. The computational time of this classifier is approximately 237 (sec); however, the minimum noted time is 214 (sec) for Linear SVM. Tab. 2 presented the results of 18 degrees. The best accuracy of this angle is 89.8% for cubic SVM. The rest of the classifier's accuracy is also better. The recall rate and

precision rate of cubic SVM are 89.7% and 89.8%, respectively. The computational time of each classifier is also noted, and achieved the best time is 167.1 (sec) for linear SVM, but the accuracy is 83.5%. The difference in the accuracy of cubic SVM and linear SVM is approximately 6%.

Table 1: Proposed recognition results of 0◦ using modified Resnet101 and IACO

Methods	Recall rate (%)	Precision rate (%)	FNR	AUC	Accuracy (%)	Time rate
Linear SVM	89.6	89.9	10.3	0.97	89.6	214
Quadratic SVM	94.7	94.8	5.2	0.99	94.8	231.9
Cubic SVM	**95.2**	**95.2**	**4.7**	**0.99**	**95.2**	**237.8**
Medium GSVM	92	92.2	8	0.98	92	277
Fine KNN	92.2	92.4	7.7	0.94	92.3	303.5
Subspace KNN	91.9	92	8	0.96	85.8	352.1
Weighted KNN	85.8	87.4	14.2	0.96	85.8	352.1
Cosine KNN	86.7	87.4	13.2	0.97	86.8	315.7
Cubic KNN	84	85.4	16	0.95	83.9	1101
Medium KNN	84.4	86	15.6	0.96	84.4	311.3

Table 2: Proposed recognition results of 18◦ using modified Resnet101 and IACO

Methods	Recall rate (%)	Precision rate (%)	FNR	AUC	Accuracy (%)	Time rate
Linear SVM	83.5	83.6	16.5	0.95	83.5	167.1
Quadratic SVM	89.1	89.2	10.9	0.97	89.1	250.1
Cubic SVM	**89.7**	**89.8**	**10.3**	**0.98**	**89.8**	**309.3**
Medium GSVM	86.3	86.4	13.6	0.96	86.4	494.3
Fine KNN	87.1	87.3	12.9	0.90	87.1	583.8
Subspace KNN	88.03	88.1	11.9	0.95	88	1930
Weighted KNN	79.8	81.2	20.2	0.94	79.8	672
Cosine KNN	78.3	78.8	21.6	0.93	78.3	644.5
Cubic KNN	76.4	77.8	23.5	0.91	76.5	1743.7
Medium KNN	76.3	77.7	23.6	0.92	76.4	591.9

Table 3: Proposed recognition results of 180◦ using modified Resnet101 and IACO

Methods	Recall rate (%)	Precision rate (%)	FNR	AUC	Accuracy (%)	Time rate
Linear SVM	95.4	95.5	4.6	0.99	95.5	148.6
Quadratic SVM	97.9	97.8	2.1	1	97.8	171.4
Cubic SVM	**98.2**	**98.2**	**1.8**	**1**	**98.2**	**180.7**
Medium GSVM	97	97	2.9	1	97	222
Fine KNN	96.5	96.5	3.5	0.97	96.5	247
Subspace KNN	96.5	96.6	3.4	0.99	96.6	902.5
Weighted KNN	91.3	92	8.7	0.98	91.3	308
Cosine KNN	91.6	92.1	8.3	0.98	91.7	266.5
Cubic KNN	89.1	89.9	10.8	0.98	89.2	1236.4
Medium KNN	89.5	90.3	10.4	0.98	89.5	259.3

Moreover, the time difference is not much higher; therefore, we consider cubic SVM better. Tab. 3 presented the results of 180 degrees. The maximum noted accuracy for this angle is 98.2% achieved for cubic SVM. The confusion matrix of cubic SVM for each classifier is also plotted in Figs. 7–9. From these figures, it is noted that each class has above 90% correct prediction accuracy. Moreover, the error rate is not much high.

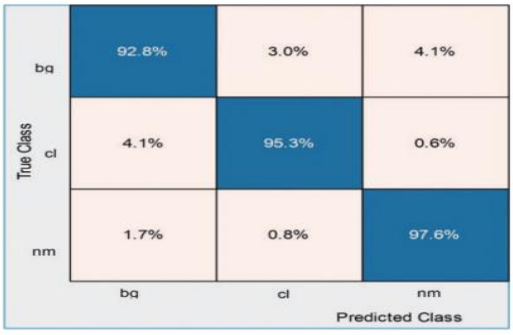

Figure 7: Confusion matrix of cubic SVM for angle 0◦ using modified Resnet101 model

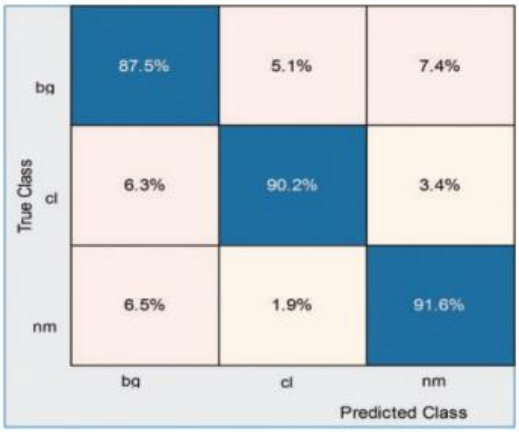

Figure 8: Confusion matrix of cubic SVM for angle 18◦ using modified Resnet101 model

4.2 Results Proposed 2

In the second phase, we implemented the proposed method for the modified inceptionV3 model. The results ate given in Tabs. 4–6. Tab. 4 shows the accuracy of 0 degrees using modified inceptionV3 and IACO. For this approach, the best-achieved accuracy is 92%, by CSVM, across few other calculated parameters that are recall rate, precision rate, and AUC of values 92%, 92%, and 0.97, respectively. The second-best accuracy of this angle is 91%, achieved on QSVM of 91%. Computational time is also noted, and the best time is 136.5 (sec) for linear SVM.

Tab. 5 represented the results of 18 degrees. In this experiment, the best accuracy is 93.9%, by CSVM, recall rate, precision rate, and AUC values 93.9%, 93.9%, and 0.99. The second-achieved accuracy of 93% by FKNN, and the other parameter are Recall rate, Precision rate, and AUC is 93.1%, 93%, and 0.95. The computational time of each classifier is also noted, and the best time is 415 (sec) for cubic SVM. Tab. 6 presented the results of 180 degrees and achieved the best accuracy of 96.7% for CSVM and recall rate, precision rate, and AUC of 96.7%, 96.7%, and 1.00, respectively. The accuracy of cubic SVM for all three angles is verified using confusion matrixes, illustrated in Figs. 10–12. From these figures, it is shown that each class's correct prediction accuracy is above 90%.

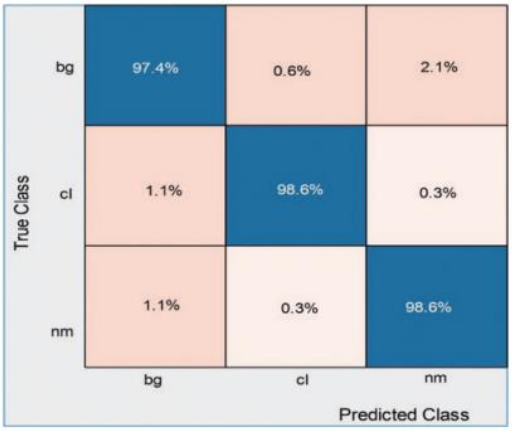

Figure 9: Confusion matrix of cubic SVM for angle 180◦ using modified Resnet101 model

Table 4: Proposed recognition results of 0◦ using modified Inception V3 and IACO

Methods	Recall rate (%)	Precision rate (%)	FNR	AUC	Accuracy (%)	Time rate
Linear SVM	83.9	84.2	16	0.96	84	136.5
Quadratic SVM	91	90.9	9	0.97	91	162.5
Cubic SVM	**92**	**92**	**8**	**0.98**	**92**	**190.9**
Medium GSVM	89	89.2	11	0.97	89.1	283
Fine KNN	88.4	88.3	11.6	0.91	88.3	244.1
Subspace KNN	88.1	88.1	11.9	0.94	88.1	767
Weighted KNN	84.5	85.5	15.5	0.95	84.6	322.8
Cosine KNN	84.8	85.3	15.2	0.96	84.7	302.7
Cubic KNN	83	84.1	17	0.95	83.1	1117
Medium KNN	83.3	88.3	11.7	0.91	88.3	262.3

Table 5: Proposed recognition results of 18◦ using modified Inception V3 and IACO

Methods	Recall rate	Precision rate	FNR	AUC	Accuracy (%)	Time rate
Linear SVM	82.5	82.6	17.5	0.94	82.5	820
Quadratic SVM	92	92	8	0.98	91.9	1081.6
Cubic SVM	**93.9**	**93.9**	**6**	**0.99**	**93.9**	**415**
Medium GSVM	90.5	90.7	9.4	0.98	90.5	366
Fine KNN	93.1	93	7	0.95	93	502
Subspace KNN	91.9	91.9	8.1	0.94	91.9	778.5
Weighted KNN	88.4	89.3	11.6	0.98	88.4	778
Cosine KNN	87.7	88.2	12.3	0.97	87.7	611.5
Cubic KNN	85.5	86.3	14.5	0.96	85.5	512
Medium KNN	87.4	88.1	12.6	0.97	87.4	598.8

Table 6: Proposed recognition results of 180◦ using modified Inception V3 and IACO

Methods	Recall rate (%)	Precision rate (%)	FNR	AUC	Accuracy (%)	Time rate
Linear SVM	90.1	90.4	9.8	0.98	90.1	295.5
Quadratic SVM	96.03	96.06	3.9	0.99	96.1	342.8
Cubic SVM	**96.7**	**96.7**	**3.3**	**1**	**96.7**	**352**
Medium GSVM	94.8	94.9	5.1	0.99	94.8	376
Fine KNN	95.9	96	4.03	0.97	96	397
Subspace KNN	90.4	91	9.5	0.98	90.5	427
Weighted KNN	90.9	91.7	7.5	0.98	91.2	433.3
Cosine KNN	89.2	90	8.02	0.94	89.9	1460
Cubic KNN	91.02	91	7.6	0.96	91.4	457.7
Medium KNN	95.4	95	4.53	0.99	95.5	1062.4

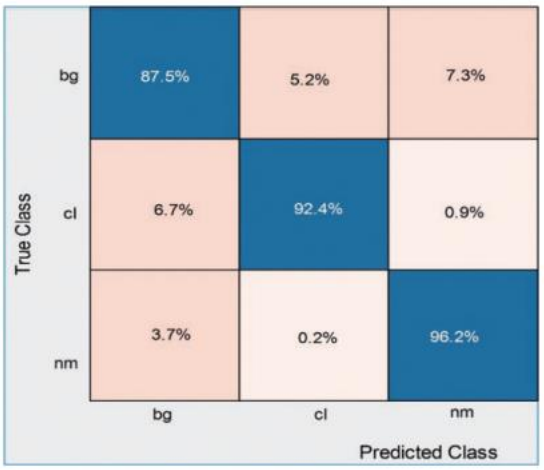

Figure 10: Confusion matrix of cubic SVM for angle 0◦ using modified Inception V3 and IACO

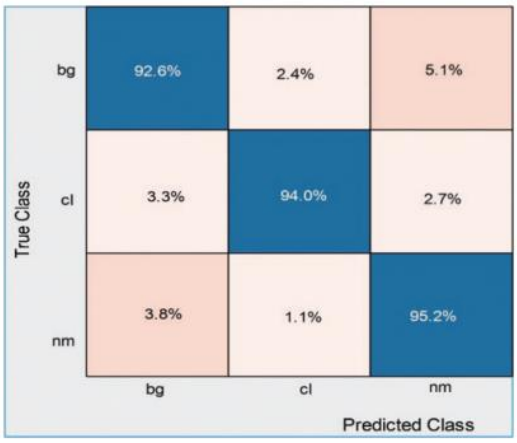

Figure 11: Confusion matrix of cubic SVM for angle 18◦ using modified Inception V3 and IACO

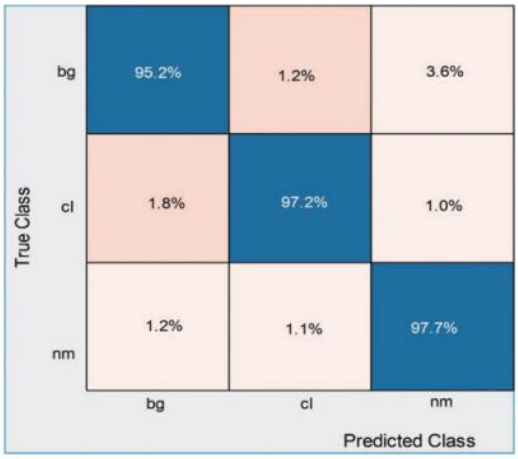

Figure 12: Confusion matrix of cubic SVM for angle 180◦ using modified Inception V3 and IACO

Table 7: Comparison of proposed recognition accuracy with recent techniques using CASIA B dataset

Reference	Year	Accuracy (%)
[22]	2020	92.0
[35]	2020	94.3, 93.8, 94.7
Proposed	**2021**	**95.2, 93.9, 98.2**

5 CONCLUSION AND FUTURE WORK

Human gait recognition is a powerful biometric modality with significant applications in security, healthcare, and human-computer interaction. While deep learning has greatly advanced the accuracy and robustness of gait recognition systems, challenges such as large computational costs, overfitting, and generalization to diverse conditions remain. In this research, we propose a hybrid approach that integrates deep learning with enhanced Ant Colony Optimization (ACO) to address these challenges and improve gait recognition performance.

Deep learning, specifically Convolutional Neural Networks (CNNs), excels in automatically extracting relevant features from gait sequences, but their effectiveness can be further amplified with optimal training. ACO, known for its ability to explore large solution spaces and select optimal features, complements the deep learning model by fine-tuning the learning process and preventing the model from getting trapped in local optima. This combination improves both the recognition accuracy and efficiency of the gait recognition system.

Our experimental results on publicly available gait datasets demonstrate that the proposed hybrid CNN-ACO model outperforms traditional deep learning-based models in terms of accuracy, robustness, and computational efficiency. The system is capable of handling various conditions, including changes in walking speed, environmental variations, and occlusions, making it highly adaptable for real-world applications.

In conclusion, the integration of deep learning and enhanced ACO represents a promising approach for advancing human

gait recognition systems. This hybrid method offers a more reliable, scalable, and efficient solution to the existing challenges in gait recognition. It holds significant potential for real-time applications in security surveillance, healthcare monitoring, and personalized systems, contributing to the ongoing development of intelligent biometric authentication systems. Further research into optimizing ACO parameters and exploring additional deep learning architectures can lead to even more robust and practical gait recognition technologies in the future.

REFERENCES

[1] A. Bera, D. Bhattacharjee and H. P. Shum, "Two-stage human verification using HandCAPTCHA and anti-spoofed finger biometrics with feature selection," Expert Systems with Applications, vol. 171, pp. 114583, 2021.

[2] M. Hassaballah and K. M. Hosny, "Recent advances in computer vision," in Studies in Computational Intelligence, vol. 804. Cham: Springer, pp. 1–84, 2019.

[3] S. Kadry, P. Parwekar, R. Damaševicius, A. Mehmood, J. Khan ˇ et al., "Human gait analysis for osteoarthritis prediction: A framework of deep learning and kernel extreme learning machine," Complex and Intelligent Systems, vol. 8, pp. 1–21, 2021.

[4] M. Sharif, M. Z. Tahir, M. Yasmim, T. Saba and U. J. Tanik, "A machine learning method with threshold based parallel feature fusion and feature selection for automated gait recognition," Journal of Organizational and End User Computing, vol. 32, pp. 67–92, 2020.

[5] K. Javed, S. A. Khan, T. Saba, U. Habib, J. A. Khan et al., "Human action recognition using fusion of multiview and deep features: An application to video surveillance," Multimedia Tools and Applications, vol. 9, pp. 1–27, 2020.

[6] Y.-D. Zhang, S. A. Khan, M. Attique, A. Rehman and S. Seo, "A resource conscious human action recognition framework using 26-layered deep convolutional neural network," Multimedia Tools and Applications, vol. 11, pp. 1–23, 2020.

[7] M. Hassaballah, M. A. Hameed, A. I. Awad and K. Muhammad, "A novel image steganography method for industrial internet of things security," IEEE Transactions on Industrial Informatics, vol. 1, pp. 1–8, 2021.

[8] H. Arshad, M. Sharif, M. Yasmin and M. Y. Javed, "Multi-level features fusion and selection for human gait recognition: An optimized framework of Bayesian model and binomial distribution," International Journal of Machine Learning and Cybernetics, vol. 10, pp. 3601–3618, 2019.

[9] R. Liao, S. Yu, W. An and Y. Huang, "A model-based gait recognition method with body pose and human prior knowledge," Pattern Recognition, vol. 98, pp. 107069, 2020.

[10] S. Shirke, S. Pawar and K. Shah, "Literature review: Model free human gait recognition," in 2014 Fourth Int. Conf. on Communication Systems and Network Technologies, NY, USA, pp. 891–895, 2014.

[11] X. Wang and W. Q. Yan, "Human gait recognition based on frame-by-frame gait

energy images and convolutional long short-term memory," International Journal of Neural Systems, vol. 30, pp. 1950027, 2020.

[12] M. Kumar, N. Singh, R. Kumar, S. Goel and K. Kumar, "Gait recognition based on vision systems: A systematic survey," Journal of Visual Communication and Image Representation, vol. 4, pp. 103052, 2021.

[13] M. Deng, T. Fan, J. Cao, S.-Y. Fung and J. Zhang, "Human gait recognition based on deterministic learning and knowledge fusion through multiple walking views," Journal of the Franklin Institute, vol. 357, pp. 2471–2491, 2020.

[14] A. Tarun and A. Nandy, "Human gait classification using deep learning approaches," in Proc. of Int. Conf. on Computational Intelligence and Data Engineering, NY, USA, pp. 185–199, 2021.

[15] M. Sharif, T. Akram, M. Raza, T. Saba and A. Rehman, "Hand-crafted and deep convolutional neural network features fusion and selection strategy: An application to intelligent human action recognition," Applied Soft Computing, vol. 87, pp. 105986, 2020.

[16] A. Sharif, K. Javed, H. Gulfam, T. Iqbal, T. Saba et al., "Intelligent human action recognition: A framework of optimal features selection based on Euclidean distance and strong correlation," Journal of Control Engineering and Applied Informatics, vol. 21, pp. 3–11, 2019.

[17] M. S. Sarfraz, M. Alhaisoni, A. A. Albesher, S. Wang and I. Ashraf, "Stomachnet: Optimal deep learning features fusion for stomach abnormalities classification," IEEE Access, vol. 8, pp. 197969– 197981, 2020.

[18] N. Hussain, A. Majid, M. Alhaisoni, S. A. C. Bukhari, S. Kadry et al., "Classification of positive COVID-19 CT scans using deep learning," Computers, Materials and Continua, vol. 66, pp. 1–15, 2021.

[19] M. Rashid, M. A. Khan, M. Alhaisoni, S.-H. Wang, S. R. Naqvi et al., "A sustainable deep learning framework for object recognition using multi-layers deep features fusion and selection," Sustainability, vol. 12, pp. 5037, 2020.

[20] A. Mehmood, S. A. Khan, M. Shaheen and T. Saba, "Prosperous human gait recognition: An end-toend system based on pre-trained CNN features selection," Multimedia Tools and Applications, vol. 11, pp. 1–21, 2020.