Computer Systems
Science and Engineering

Tech Science Press

DOI: 10.32604/csse.2023.040410
Article

Suspicious Activities Recognition in Video Sequences Using DarkNet-NasNet Optimal Deep Features

Safdar Khan[1], **Muhammad Attique Khan**[2], **Jamal Hussain Shah**[1,*], **Faheem Shehzad**[2], **Taerang Kim**[3] **and Jae-Hyuk Cha**[3]

[1]Department of Computer Science, COMSATS University Islamabad, Wah Campus, Wah Cantt, 47040, Pakistan
[2]Department of Computer Science, HITEC University, Taxila, 47080, Pakistan
[3]Department of Computer Science, Hanyang University, Seoul, 04763, Korea
*Corresponding Author: Jamal Hussain Shah. Email: jhshah@ciitwah.edu.pk
Received: 16 March 2023; Accepted: 18 May 2023; Published: 28 July 2023

Abstract: Human Suspicious Activity Recognition (HSAR) is a critical and active research area in computer vision that relies on artificial intelligence reasoning. Significant advances have been made in this field recently due to important applications such as video surveillance. In video surveillance, humans are monitored through video cameras when doing suspicious activities such as kidnapping, fighting, snatching, and a few more. Although numerous techniques have been introduced in the literature for routine human actions (HAR), very few studies are available for HSAR. This study proposes a deep convolutional neural network (CNN) and optimal features-based framework for HSAR in video frames. The framework consists of various stages, including preprocessing video frames, fine-tuning deep models (Darknet 19 and Nasnet mobile) using transfer learning, serial-based feature fusion, feature selection via equilibrium feature optimizer, and neural network classifiers for classification. Fine-tuning two models using some hit and trial methods is the first challenge of this work that was later employed for feature extraction. Next, features are fused in a serial approach, and then an improved optimization method is proposed to select the best features. The proposed technique was evaluated on two action datasets, Hybrid-KTH01 and Hybrid-KTH02, and achieved an accuracy of 99.8% and 99.7%, respectively. The proposed method exhibited higher precision compared to existing state-of-the-art approaches.

Keywords: Suspicious activity; deep learning; weights; surveillance; optimization

1 Introduction

Human suspicious activity recognition (HAR) is becoming popular in machine learning (ML) and artificial intelligence (AI) [1,2]. Motivated by the vast range of applications, major advances in learning and recognizing human actions by utilizing different techniques of computer vision (CV)

[3,4]. HAR can be used in different medical and biometric domains such as public surveillance, airports, shopping malls, home care for old and children, computer gaming, human-robot and human-computer interaction (HBI/HCI), and video retrieval [5–7]. To design an intelligent human–computer interaction, a system must perceive human motions and interpret their actions and intentions. HAR uses range sensors, cameras, and other sensors to recognize and evaluate human activities or actions in real time [8]. Thus, many tasks would be transformed if computer systems could automatically understand daily human behavior. Depending on intricacy, human movements can be classified as actions, gestures, or interactions [6,7]. Most available action recognition (AR) algorithms are mostly based on RGB data. Nevertheless, HAR based on RGB data faces the following challenges: (i) Complex backdrop, image shadows, occlusion, and varying light conditions, and size and scale change will create significant recognition challenges, which is also true for RGB-based AR [9]. (ii) The same event will yield distinct perceptions from various vantage points. (iii) Same action done by multiple individuals will vary substantially, and two distinct types of actions may share many similarities. These intrinsic flaws of RGB visual information will hinder HAR based on RGB data [10].

Deep learning has recently demonstrated superior performance in ML and CV for various applications, including agricultural, medical, video surveillance, biometrics, detection, and classification of different objects [11]. Even if features are retrieved automatically by deep learning, the accuracy of action detection has risen compared to handcrafted action features. To increase the recognition rate of HAR, CNN deep networks are utilized. There are two main steps of HAR techniques: extracting deep features and then classifying the actions. For HAR, pretrained deep CNN networks such as Resnet [12], Inceptionv3 [13], Alex net [14], Exception [15], VGG [16], Efficientnet [17], and other deep networks are utilized with the concept of transfer learning (TL) [14]. Even if features are retrieved automatically by deep learning, the accuracy of action detection has risen compared to handcrafted action features [18]. Due to complex and large datasets, we sometimes used more than one CNN model to increase the recognition accuracy with the fusion process because a single network is not working well on these datasets. After the fusion process, many features are obtained that decrease the accuracy and increase the computation time. So, this issue is resolved by different feature optimization techniques such as ant colony, variance-based optimization, firefly optimizer, whale optimizer, tree-based optimizer, and a few others [19]. The feature selection process involves identifying and selecting a smaller set of important features or variables from a larger pool of options to enhance the accuracy and efficiency of a machine learning model. The goal of this technique is to simplify the model and remove irrelevant or redundant features that may cause overfitting and reduce overall performance. Choosing the appropriate feature selection method relies on the particular dataset and problem at hand, and a blend of various techniques might be utilized for optimal results. As a crucial phase of the machine learning process, feature selection can produce models that are simpler and more easily understood, while also demonstrating enhanced generalization and quicker training times. Optimal feature vector is finally pass to the different ML and NN classifiers such as K-Nearest Neightbor (KNN), Tree, support vector machine (SVM), Ada boost, linear discriminant analysis (LDA), and Neural Network classifier for action classifications [20]. The major contributions of this work are as follows:

– Two pre-trained deep neural networks are fine-tuned using a hit-and-trial method. In this, a few layers from the middle was skipped and then added at the end.
– Models are trained with static hyperparameters, and extracted features are fused in a serial-based mathematical formulation instead of simple concatenation.
– Propose an improved feature selection technique called Equilibrium State Static Values.

2 Related Work

Numerous techniques for recognizing activities have been suggested by researchers, utilizing both handcrafted features and features generated through deep learning [21,22]. Jiang et al. [23] present a HAR technique in which a fusion is performed on temporal and enhanced features (TEFE). The temporal feature extraction technique is used to get the short- and long-term features. To reduce the parameters in bilinear pooling. V1 and V2 datasets and jester datasets are used to evaluate the model. Accuracy of 96.8% on the jester dataset and 88.7% on something V2 dataset is achieved. Yang et al. [24] suggested the STA-TSN approach for AR. This technique is used on long videos and gets the best features from time and space data. Four action datasets, such as thumos14, jhmdb, hmdb51, and ucf101 are used to evaluate the proposed framework. Attained 78.4% accuracy on jhmdb, 67.9% on hmbd51%, and 93.3% on the ucf101 dataset. Shi et al. [25] suggested a pose-based method for HAR. In this work, a light weighted graph network is employed with dense intermediate supervision. Three datasets such as NTU-RGBD, Penn action, sub-jhmdb are used to evaluate the suggested framework. Attained the accuracy of 96.4%, 99%, and 89.3%, respectively.

Yang et al. [26] proposed a Hybrid Net model for AR, combining CNN and GCN networks to explain the body joints well. The suggested work tested on three action datasets: Skeleton-Kinetics, NTU-RGBD 120, and NTU-RGBD datasets, attained accuracies of 62.3%, 96.9%, and 89.0%, respectively. Shen et al. [27] present a framework for HAR containing two modules. In the first module, a complex network extracts the features. The extracted feature vector is fused with the LSTM to recognize the second module's actions efficiently. UTKinect-Action3D, NTU RGB + D60, and MSR Action3D datasets are utilized to test the presented method. Achieved accuracies of 90.4%, 91.8%, and 90.5%, respectively. Liu et al. [28] presented a two-stream model based on spatial, temporal learning interaction (STLIT) to recognize human activities. This work is based on two different streams: temporal and spatial streams. An interactive connection is established between both streams by using the proposed framework. They tested their work on three activity datasets such as kinetics, HMDB51, and UCF101. They attained an accuracy of 72.4%, 72.1%, and 95.6%, respectively. Afza et al. [29] proposed a technique combining selected and extracted features, utilizing M-SVM to identify actions. The effectiveness of this approach was evaluated on multiple datasets, including Weizmann, UCF YouTube, KTH, and UCF Sports, achieving high levels of accuracy ranging from 94.5% to 100%. The authors [1] introduced an innovative framework for activity recognition that utilizes features obtained from RGB-D information through a complex network. They utilized a meta-path concept with this network to detect unusual actions. The effectiveness of this approach was assessed on two datasets, MSR Activity-3D, and MSR-Action, achieving accuracies of 96% and 98%, respectively. The authors [10] proposed a selective ensemble SVM approach for activity recognition, which fuses multi-model features including (HJF), HOG, (DMM-LBP) features. Their method was evaluated on the two datasets (CAD 60 and G3D), achieving 92.1% and 92% accuracy, respectively. Liu et al. [30] suggested a (KA-AGTN) network for activity recognition. The proposed work tested on three datasets: Kinetics-Skeleton 400, NTU-RGBD 120, and NTU-RGBD 60. They achieved accuracies of 61%, 88.0%, and 96.1%, respectively. Shehzad et al. [31] present two steam CNN for HAR. In this work, two CNN models are used for deep features, and IWOA selection algorithm is utilized to extract the best feature set. Four activity datasets, namely Hollywood, Ut-interaction, UCF Sports, and IXMAS datasets, are used to evaluate the suggested method, achieving accuracies of 99.1%, 100%, 100%, and 99.9%, respectively. The authors [32] proposed a technique for activity recognition using CNN and Bi-Longest shortest memory (BLSTM) network. They also suggested a swarm intelligence-based algorithm for selecting the best hyperparameters for neural networks. The effectiveness of this approach was evaluated using UCF50, KTH, and UCF101 datasets, and resulted in improved performance.

3 Proposed Methodology

Fig. 1 illustrates the proposed AR technique, which comprises several steps. These steps involve initial preprocessing of video frames, fine-tuning a darknet 19 and Nasnet mobile deep model through transfer learning, extracting features, fusing the feature vectors using a modified correlation serial-based method, selecting features through equilibrium optimizer, and performing classification using various neural network classifiers. Finally, the trained classifiers' results are evaluated at the end of the process. Each step is elaborated on detail in the subsequent sections.

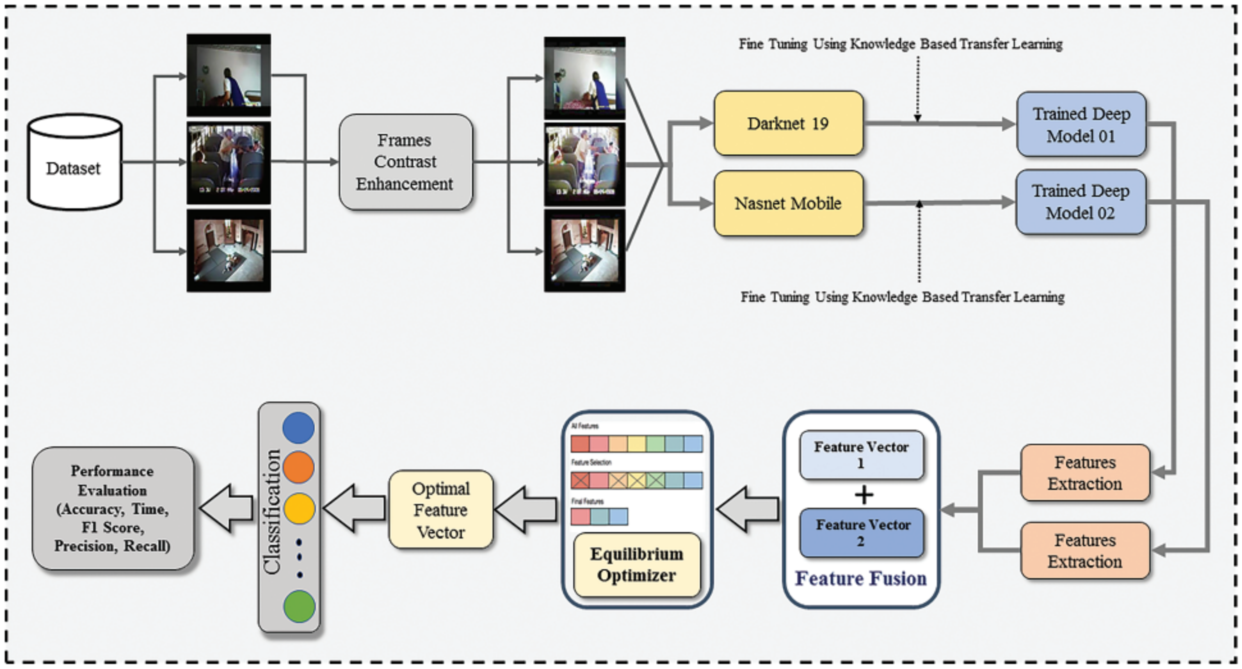

Figure 1: Proposed method for action recognition (AR)

3.1 Frames Initial Preprocessing

Video frame preprocessing involves using various methods to modify, filter, or enhance video frames before being analyzed or processed by a machine learning or computer vision algorithm. This preparatory step can enhance the precision and efficiency of the analysis by decreasing noise, eliminating extraneous data, and highlighting relevant features. The preprocessing phase in this study aims to convert action video sequences into individual frames. Initially, the dimensions of each video frame were $512 \times 512 \times k$, where k equals 3. However, the frames are subsequently transformed by scaling them down to $256 \times 256 \times 3$ pixels.

To enhance the image quality, a sub-image dualistic histogram equalization (SDHE) technique is performed [33]. SDHE has a major advantage over traditional histogram equalization as it produces superior results for images containing both dark and bright regions. This is achieved by applying histogram equalization to each sub-image separately, which enhances the contrast in dark and bright regions without causing over-brightening or over-darkening. As a result, SDHE can significantly improve the overall contrast and quality of such images while preserving their original characteristics. SDHE is a technique specifically designed to improve the contrast of images that contain dark and bright regions. It is a variation of histogram equalization, where the image is divided into smaller sub-images, and each sub-image is subjected to histogram equalization individually. After the sub-images are processed, they are recombined to generate the final image with improved contrast.

Let *y* is an image, and the gray level of that image is $y = y_f$. Sub-images are denoted by y_m and y_n. Center pixel index is denoted by f.

$$y = y_m \cup y_n \tag{1}$$

$$y_m = \left\{y\,(i.j)\,|y\,(i.j)\, < y_f, \forall y\,(i.j) \in y\right\} \tag{2}$$

$$y_n = \left\{y\,(i.j)\,|y\,(i.j) \geq y_f, \forall y\,(i.j) \in y\right\} \tag{3}$$

Upper transformation uses for less bright images.

$$y_m = \left\{y_0,\ y_1, \ldots\ldots\ldots\ldots y_{f-1}\right\} \tag{4}$$

$$y_n = \left\{y_f,\ y_{f+1}, \ldots\ldots\ldots\ldots y_{m-1}\right\} \tag{5}$$

Aggregation of grey level original image as follows

$$\left\{Q_{0,}\, Q_1\,, \ldots\ldots\ldots\ldots Q_{f-1}\right\} \tag{6}$$

$$\left\{Q_{f,}\, Q_{f+1}, \ldots\ldots\ldots\ldots Q_{m-1}\right\} \tag{7}$$

Suppose

$$p = \sum\nolimits_{i=0}^{f-1} Q_i \tag{8}$$

$$p = \sum\nolimits_{i=f}^{m-1} Q_i \tag{9}$$

$$Q\,(y_m) = \frac{p_i}{Q_m}\ ,\ i = 0, 1, 2, \ldots\ldots\ldots\ldots, f - 1 \tag{10}$$

$$Q\,(y_n) = \frac{p_i}{Q_n}\ ,\ i = f, f + 1, f + 2, \ldots\ldots\ldots\ldots, m - 1 \tag{11}$$

Now for commulative distribution function (CDF) is formulated as follows:

$$D_m\,(y_k) = \frac{1}{Q}\sum\nolimits_{i=0}^{f-1} Q_i \tag{12}$$

$$D_n\,(y_k) = \frac{1}{Q}\sum\nolimits_{i=f}^{m-1} Q_i \tag{13}$$

Transformation function

$$f_m\,(y) = y_0 + \left(y_{f-1} - y_0\right) \times\ D_m\,(y_k) \tag{14}$$

$$f_n\,(y) = Y_f + \left(y_{m-1} - y_f\right) \times\ D_n\,(y_k) \tag{15}$$

Output image

$$X = f_m(y) \cup f_n(y) \tag{16}$$

3.2 Deep CNN Model

CNN, or Convolutional Neural Network, is a type of neural network architecture commonly employed in computer vision and image recognition tasks. The network is constructed using various layers, each with a specific function, to extract meaningful features from the input data. CNN has at least one convolutional layer in its layers. There are three basic layers in the CNN network. The convolutional layer performs feature extraction by applying filters or kernels to the input image. Each filter is convolved across the entire input image, computing element-wise multiplication and summing up the results to generate a single output. The output of this layer is a collection of feature maps that capture distinct features of the input image. The convolutional operation can be expressed mathematically as follows:

$$A[i \times j] = (m * n)[i \times j] = \sum_l \sum_k n[l,\ k]\, m[i - l, j - k] \tag{17}$$

where the input image is 'm' and kernel is 'n'. i × j are the number of rows and columns, and 'A' is output image.

In a CNN, the activation layer is responsible for applying a non-linear activation function to the output feature maps of the previous convolutional layer. Common activation functions include Rectified Linear Units (ReLU) and others. The purpose of this layer is to introduce non-linearity in the network, allowing it to capture more complex features from the input data. It can be expressed mathematically as follows:

$$ReLu\ Function = Max\ (0,\ y),\ y\ \in\ J \tag{18}$$

The (Max/Avg) pooling layer reduces the spatial dimensionality of the feature maps generated by the preceding convolutional layer by performing a pooling operation on each feature map, like average pooling or max pooling. This down-sampling helps make the network more efficient by reducing the parameters and computations required for processing the feature maps.

The FC layer establishes connections between all neurons in the preceding layer to every neuron in the next layer. This layer is typically used in the final stages of the network to perform the classification. Mathematically as follows:

$$W_0^{out} = J[m \times n] \tag{19}$$

$$W_l^{in} = W_{l-1}^{out} * J_l + n_l \tag{20}$$

$$W_l^{out} = \Delta_l\left(W_l^{in}\right) \tag{21}$$

In this Equation, the final fully connected (FC) layer is denoted as W_l^{out}. The symbol Δ represents the activation function, and l refers to the number of layers in question. Finally, the SoftMax layer is used for classification.

$$SoftMax\left(W_l^{out}\right) = \frac{exp(W_l^{out})}{\sum_i W_i^{out}} \tag{22}$$

3.3 Deep Feature Extraction

For feature extraction, two pre-trained models (Darknet-19 and Nasnet-Mobile) were utilized. Darknet-19 is an architecture for deep CNN intended for computer vision tasks such as object detection and classification. It is a more compact variant of the Darknet architecture developed by Joseph Redmon, with a total of 19 layers, and is consequently called Darknet-19. The Darknet-19 structure consists of 19 layers, comprising 18 convolutional layers and one fully connected layer. It employs the same fundamental building blocks as the bigger Darknet-53 architecture but has fewer layers, making it more efficient computationally and faster to train. In 2018, Google researchers introduced Nasnet Mobile, a type of CNN architecture created specifically for deployment on mobile devices. Nasnet Mobile is a convolutional neural network architecture that is specifically designed for mobile devices, with the advantage of having a relatively small number of parameters compared to other advanced CNN architectures. This feature makes it suitable for deployment on mobile devices with limited computational resources. The architecture is composed of multiple blocks of convolutional layers and pooling layers, followed by fully connected layers for classification. Additionally, it implements several optimization techniques, such as batch normalization, dropout, and learning rate scheduling to enhance the network's generalization performance.

These models were originally trained on the ImageNet dataset, which consists of 1000 distinct classes. The final fully connected layer was eliminated and replaced with new dense layers to make these models suitable for use with the Hybrid-KTH datasets. The modified models were then fine-tuned using transfer learning techniques on these datasets. A ratio of 70:30 was used for training and testing the dataset, respectively. The hyperparameters were configured with a mini-batch size of 16, an initial learning rate of 0.01, 200 epochs, and a dropout of 0.3. Finally, the new deep model fine-tuned with transfer learning was trained. Fig. 2 shows the TL process for AR.

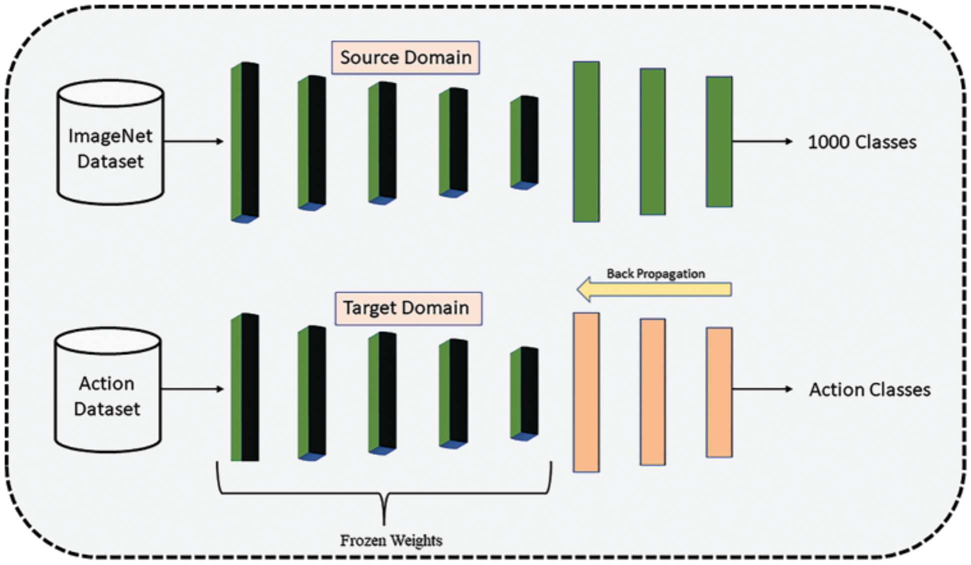

Figure 2: Knowledge-based transfer learning process for AR

2-D Global Average Pooling (2gap) is used to extract the features from the Darknet-19 model. The extracted feature vector is (N × 1024) and represented with V1. For Nasnet-Mobile, the "Global Average Pooling 2d layer" (gap2) is used to extract the features. The extracted feature vector is (N × 1056) and represented with V2.

3.4 Feature Fusion

Feature fusion is a technique that involves merging feature maps or representations derived from multiple sources or modalities, aiming to enhance a model's performance. In this method, we utilized a modified correlation extended serial approach to combine the feature vectors Vec1 and Vec2. Let us assume that V1 belongs to X_a and V2 belongs to Y_b. The following formula determines the correlation between a and b:

$$Corelation = \frac{\sum\left(X_a - \overline{X}\right)\left(Y_b - \overline{Y}\right)}{\sqrt{\sum\left(X_a - \overline{X}\right)^2\left(Y_b - \overline{Y}\right)^2}} \tag{23}$$

After applying the formula, features with positive correlation are stored in a new vector called V3, while those with zero or negative correlation are stored in V4. The mean value of V4 is then computed and each feature is compared to this value using the following approach:

$$CT = \begin{cases} \widetilde{V}_4 & for\ V_4 \geq \mu \\ ignore, & otherwise \end{cases} \tag{24}$$

Finally, V5 is obtained after fusion is performed on V4 and V3 by the following formula:

$$V5\left(k\right) = \begin{pmatrix} V_3\left(k\right)_{u\times v} \\ \widetilde{V}_4\left(k\right)_{u\times v} \end{pmatrix} \tag{25}$$

The final vector V5 having (N × 2080) dimension.

Now the V5 pass to the feature selecting algorithm (Equilibrium Optimizer) to select the best features.

3.5 Feature Optimization

Feature optimization refers to pinpointing and picking out the most significant features or variables in a dataset that are relevant to the problem being addressed. The objective of feature selection is to decrease the dataset's dimensionality by eliminating insignificant, duplicated, or noisy features while preserving those most informative and capable of making accurate predictions.

In our feature selection approach, we employed the equilibrium optimization (EO) algorithm [34]. The decision to use the EO algorithm was based on its ability to bypass local minima. Furthermore, this algorithm is known to perform exploratory searches during the initial iterations and exploitative searches during the final iterations. Nevertheless, improving its exploratory and exploitative capabilities can enhance the EO algorithm's convergence towards global optima. This optimizer is applied to a fused vector V5, and after the optimization process, we obtain optimal vector V6. Below is an outline of the optimization process.

3.6 Equilibrium Optimizer

The underlying principle behind the EO algorithm is based on the dynamic mass balance concept applied to a control volume. As shown in Eq. (26), the generalized mass balance equation serves as a foundation for the EO algorithm.

$$D_v \frac{dn}{dt} = Sm_{eq} - Sm + n_g \tag{26}$$

In Eq. (26), D_v denotes the control volume with a mass of m. The expression $\frac{dn}{dt}$ represents the variation of concentration over time within the control volume D_v. The volumetric flow rate for D_v is represented by S. The equilibrium mass of m_{eq} and the mass generated inside the control volume, n_g, are also included in the Equation. According to Eq. (26), the rate of change in mass over time within the control volume is equal to the concentration produced inside the volume minus the mass that has flowed out of it. When the system reaches a state of equilibrium, the left-hand side of Eq. (26), which represents changes in mass within the control volume, becomes zero. In essence, Eq. (26) clearly represents how mass is balanced within the control volume under dynamic conditions. Eq. (1) can be transformed into Eq. (27) by rearranging and integrating it over time in order to calculate the mass present within the control volume.

$$n = n_{eq} \left(n_o - n_{eq}\right) F + \frac{R}{\gamma D_v} (1 - F) \tag{27}$$

F is calculated by following equation.

$$F = d_1 sign\,(rand - 0.5)\,[e^{-\gamma t} - 1] \tag{28}$$

In this context, the constant d1 is introduced to enhance the exploration competence of the EO algorithm. The term "sign(rand-0.5)" plays a role in determining the direction of exploration and exploitation. The turnover rate γ is utilized to determine the variation in mass, with lower values of γ resulting in higher variations and vice versa. Additionally, the variable t represents time, expressed as a function of the iteration count, denoted as "itr" and specified by Eq. (29).

$$t = \left(1 - \frac{itr}{max - itr}\right)^{\left(d2 \frac{itr}{max-itr}\right)} \tag{29}$$

The current iteration is represented by 'itr', while 'max-itr' indicates the maximum number of iterations permitted. 'd2' is a constant used to manage EO's exploitation capabilities. To improve the exploitation phase of EO, the 'Rg' (generation rate) defined in Eq. (27) is utilized as described in Eq. (30).

$$R_g = R_{g0} F \tag{30}$$

R_{g0} is calculated by following equation:

$$R_{g0} = DQ \left(n_{eq} - \gamma n\right) \tag{31}$$

In this context, the control parameter DQ is utilized to update a certain parameter, Up, by using its associated probability. The value of DQ can be obtained through Eq. (32).

$$DQ = \left\{ \begin{Bmatrix} 0.5rand & \left| rand \geq U_p \right| \\ 0 & \left| else \right| \end{Bmatrix} \right\} \tag{32}$$

The core of the EO utilizes Eq. (27) to update the mass of each particle in every iteration. The EO begins by generating an initial population randomly, which consists of a random number of particles with their corresponding masses assigned randomly according to Eq. (33).

$$n_i^{intial} = n_{min} + rand \times (n_{max} - n_{min}) \tag{33}$$

In this context, the variables $n_{\min}$ and $n_{\max}$ represent the minimum and maximum allowable masses, respectively. The function Rand generates a random number within the range of 0 to 1. The top four particles, as well as their average, are chosen to form an equilibrium pool using Eq. (34).

$$n_{eq-pool} = \{n_{eq1}, n_{eq2}, n_{eq3}, n_{eq4}, n_{eq1(ave)}\} \tag{34}$$

To update the mass of particles in each iteration, Eq. (27) is utilized based on a randomly chosen value from the equilibrium pool obtained. The selection of each particle in the pool is done with equal probability. The utilization of a pool instead of a single best particle increases the exploration of the solution space by the algorithm. The average value in the pool also enhances the exploitation of the search space. This procedure is repeated for a maximum number of iterations, max-itr, to achieve an optimized solution.

As the last step, the optimized vector V6 is passed to five distinct neural network classifiers to perform the task of classification.

4 Results

This section presents a comprehensive description of the results and their analysis. The proposed technique was evaluated on two distinct: Hybrid KTH01, and Hybrid KTH02 datasets. Several hyperparameters were used to train the neural networks, including a minibatch size of 16, a learning rate of 0.001, 200 epochs, 10-fold cross-validation, and for learning rate optimization the Adam optimizer. The dataset was split into a training set and a testing set in a 70:30 ratio. Five classifiers were used to classify the actions, including Wide Neural Network (WNN), Narrow Neural Network (NNN), Medium Neural Network (MNN), Tri-layered Neural Network (TLNN), and Bi-layered Neural Network (BLNN). The proposed technique was implemented in MATLAB 2022 on a system with a Core-i7 processor, 8 GB RAM, and a 2 GB graphics card.

4.1 Datasets Detail

Hybrid-KTH 01 Dataset: This dataset consist of 12 different action classes namely: Abuse (261 images), Arrest (500 images), Arson (250 images), Boxing (3016 images), Clapping (3700 images), Jogging (4386 images), Running (3124 images), Shoplifting (220 images), Stealing (180 images), Vandalism (180 images), Walking (3306 images), and Waving (3201 images). These classes contain total number of 22,324 images. A few sample images are shown in Fig. 3.

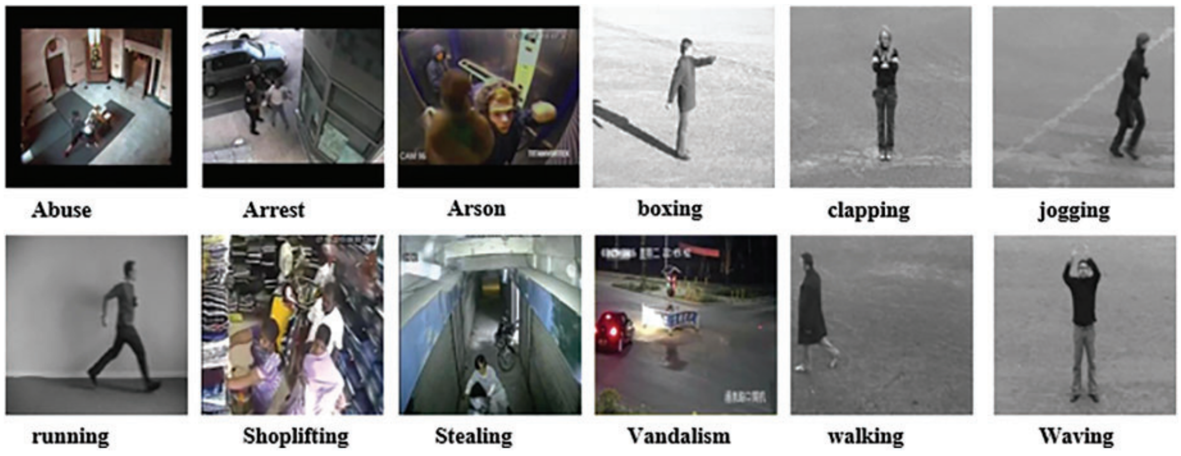

Figure 3: Sample images of Hybrid-KTH01 dataset

Hybrid-KTH 02 Dataset: This dataset consist of 13 different action classes namely: Assault (190 images), Boxing (3016 images), Burglary (190 images), Clapping (3700 images), Explosion (90 images), Fighting (150 images), Jogging (4386 images), Road Accidents (160 images), Robbery (170 images), Running (3124 images), Shooting (240 images), Walking (3306 images), and Waving (3201 images). These classes contain total number of 21,923 images. A few sample images are shown in Fig. 4.

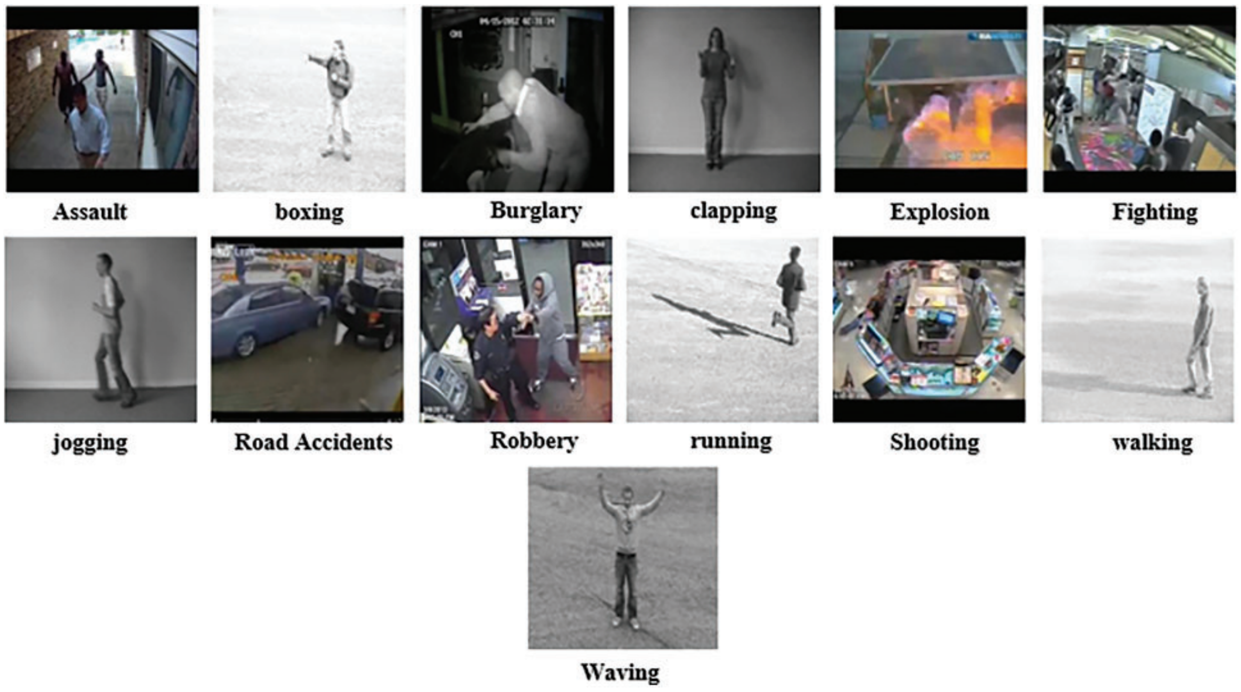

Figure 4: Sample images of Hybrid-KTH02 dataset

4.2 *Numerical Results*

The table and confusion matrices display the results of the proposed technique. Each dataset is evaluated separately, and its outcomes are presented. Hybrid-KTH01 Dataset Results: In Table 1, the outcomes of the Hybrid-KTH01 dataset are presented. By analyzing Nasnet-Mobile deep features, the Wide Neural Network (WNN) classifier obtained the highest accuracy of 98.6%. The other classifiers also demonstrated high accuracy (>98%) along with Precision, F1 score, and Recall values shown in Table 1. The confusion matrix of the WNN classifier can be used to validate the model's performance. Fig. 5 shows the class-based confusion metric. Table 1 also provides the computation time for each classifier.

In Table 2, the outcomes of the Hybrid-KTH01 dataset are presented. By analyzing Darknet-19 deep features, the Wide Neural Network (WNN) classifier obtained the highest accuracy of 99.7%. The other classifiers also demonstrated high accuracy (>98%) along with Precision, F1 score, and Recall values shown in Table 2. The confusion matrix of the WNN classifier can be used to validate the model's performance. Fig. 6 shows the class-based confusion metric. Table 2 also provides the computation time for each classifier.

Table 1: Classification results on Hybrid-KTH01 dataset using Nasnet Mobile deep features

Classifier	Parameters				
	Recall	Precision	F1 score	Accuracy (%)	Time (sec)
NNN	0.9837	0.9712	0.9775	98.2	244
MNN	0.9816	0.98	0.9816	98.4	261
WNN	0.9883	0.9866	0.9866	98.6	349
BLNN	0.9837	0.9712	0.9775	98.2	354
TLNN	0.9825	0.9808	0.98	98.3	363

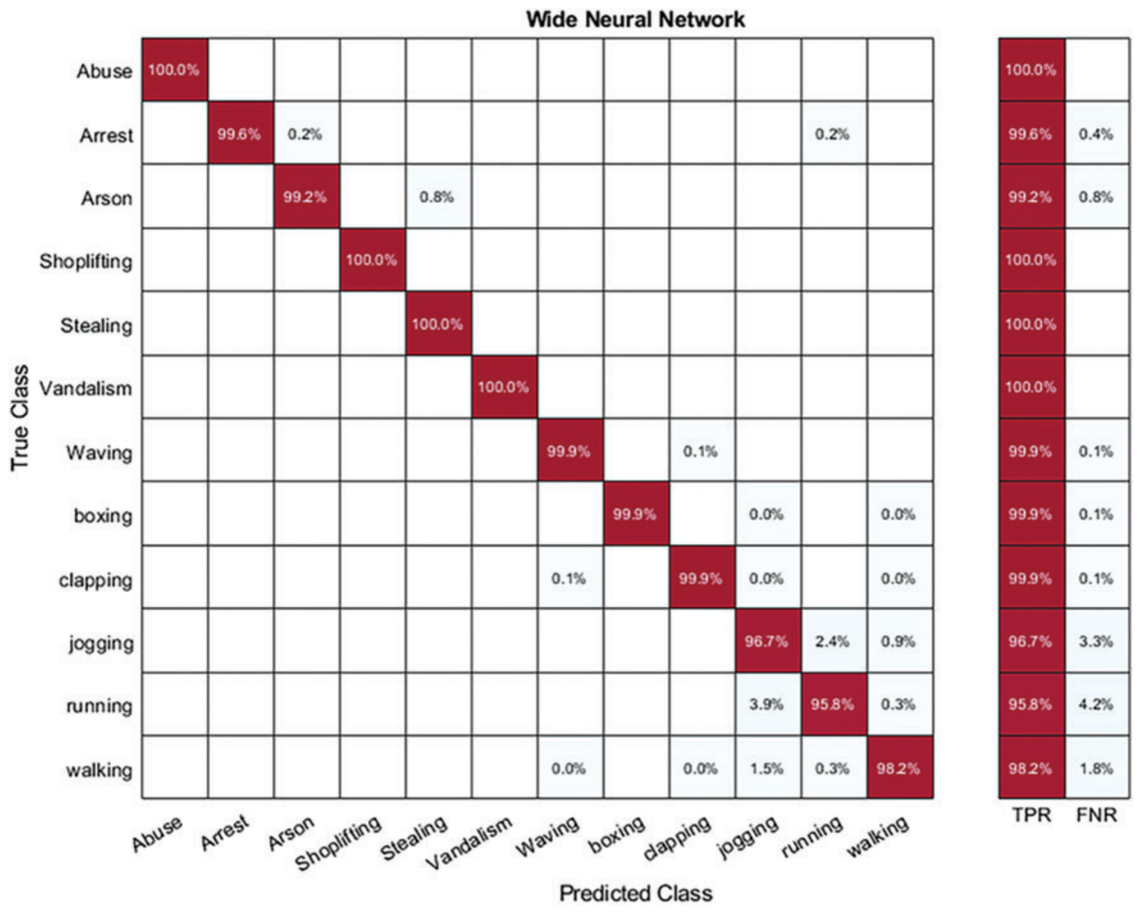

Figure 5: Class based confusion matrix of WNN classifier

Table 2: Classification results on Hybrid-KTH01 dataset using DarkNet19 deep features

Classifier	Parameters				
	Recall	Precision	F1 score	Accuracy (%)	Time (sec)
NNN	0.9983	0.9983	0.9966	99.7	48
MNN	0.9983	0.9983	0.9966	99.7	34
WNN	0.9983	0.9983	0.9966	99.7	46
BLNN	0.9883	0.9866	0.9866	98.9	75
TLNN	0.9862	0.9775	0.9825	98.5	293

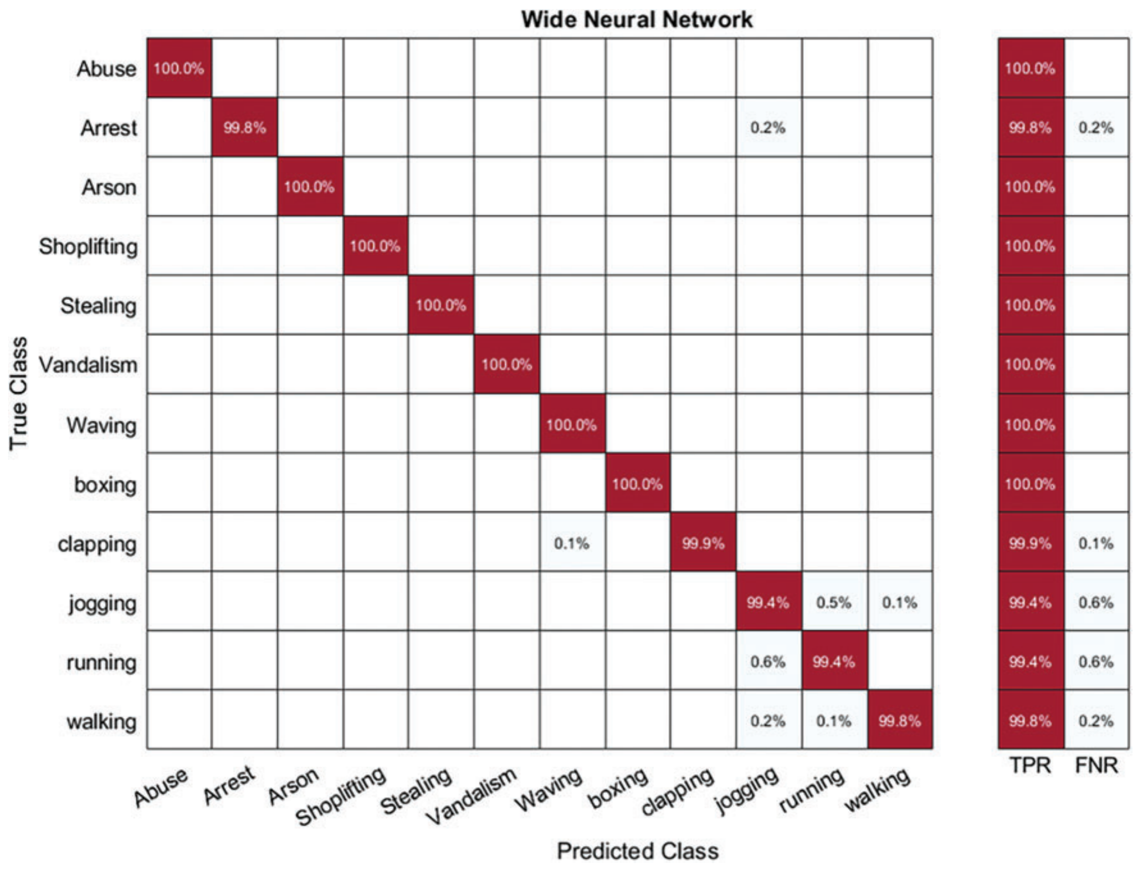

Figure 6: Class based confusion matrix of WNN classifier

In Table 3, the outcomes of the Hybrid-KTH01 dataset are presented. By analyzing fused features, the Medium Neural Network (MNN) classifier obtained the highest accuracy of 99.8%. The other classifiers also demonstrated high accuracy (>99%) along with Precision, F1 score, and Recall values shown in Table 3. The confusion matrix of the MNN classifier can be used to validate the model's performance. Fig. 7 shows the class-based confusion metric. Table 3 also provides the computation time for each classifier.

Table 3: Classification results on Hybrid-KTH01 dataset using fused features

Classifier	Parameters				
	Recall	Precision	F1 score	Accuracy (%)	Time (sec)
NNN	0.9983	0.9983	0.9966	99.7	74
MNN	1.0	1.0	1.0	99.8	77
WNN	1.0	1.0	1.0	99.8	108
BLNN	1.0	1.0	1.0	99.8	146
TLNN	0.9937	0.99	0.9937	99.4	1149

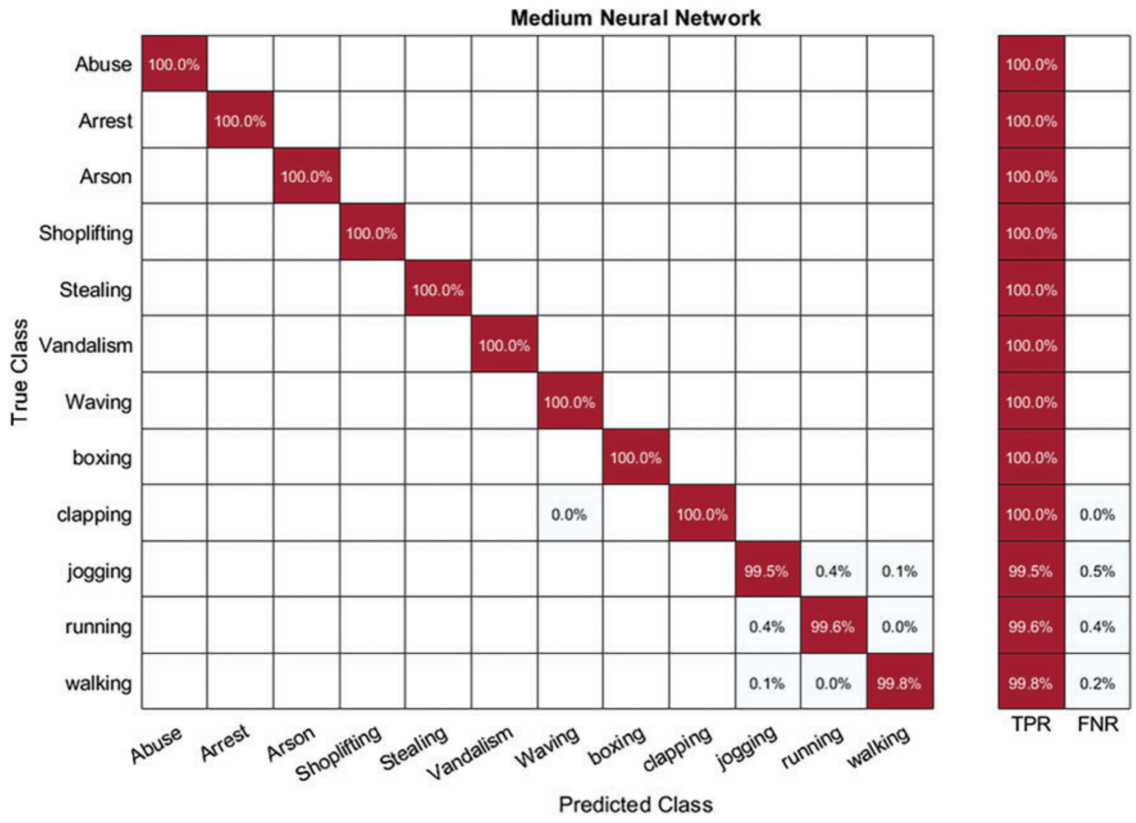

Figure 7: Class based confusion matrix of MNN classifier

Hybrid-KTH01 Dataset Results: In Table 4, the outcomes of the Hybrid-KTH01 dataset are presented. By analyzing optimized features, the Medium Neural Network (MNN) classifier obtained the highest accuracy of 99.8%. The other classifiers also demonstrated high accuracy (>99%) along with Precision, F1 score, and Recall values shown in Table 4. The confusion matrix of the MNN classifier can be used to validate the model's performance. Fig. 8 shows the class-based confusion metric. Fig. 9 shows the comparison of time of all NN classifiers. Table 4 also provides the computation time for each classifier.

Table 4: Classification results on Hybrid-KTH01 dataset using optimized features

Classifier	Parameters				
	Recall	Precision	F1 score	Accuracy (%)	Time (sec)
NNN	1.0	1.0	1.0	99.8	13
MNN	1.0	1.0	1.0	99.8	11
WNN	1.0	1.0	1.0	99.8	15
BLNN	0.9966	0.9958	0.9966	99.5	48
TLNN	0.9883	0.9866	0.9866	99.0	65

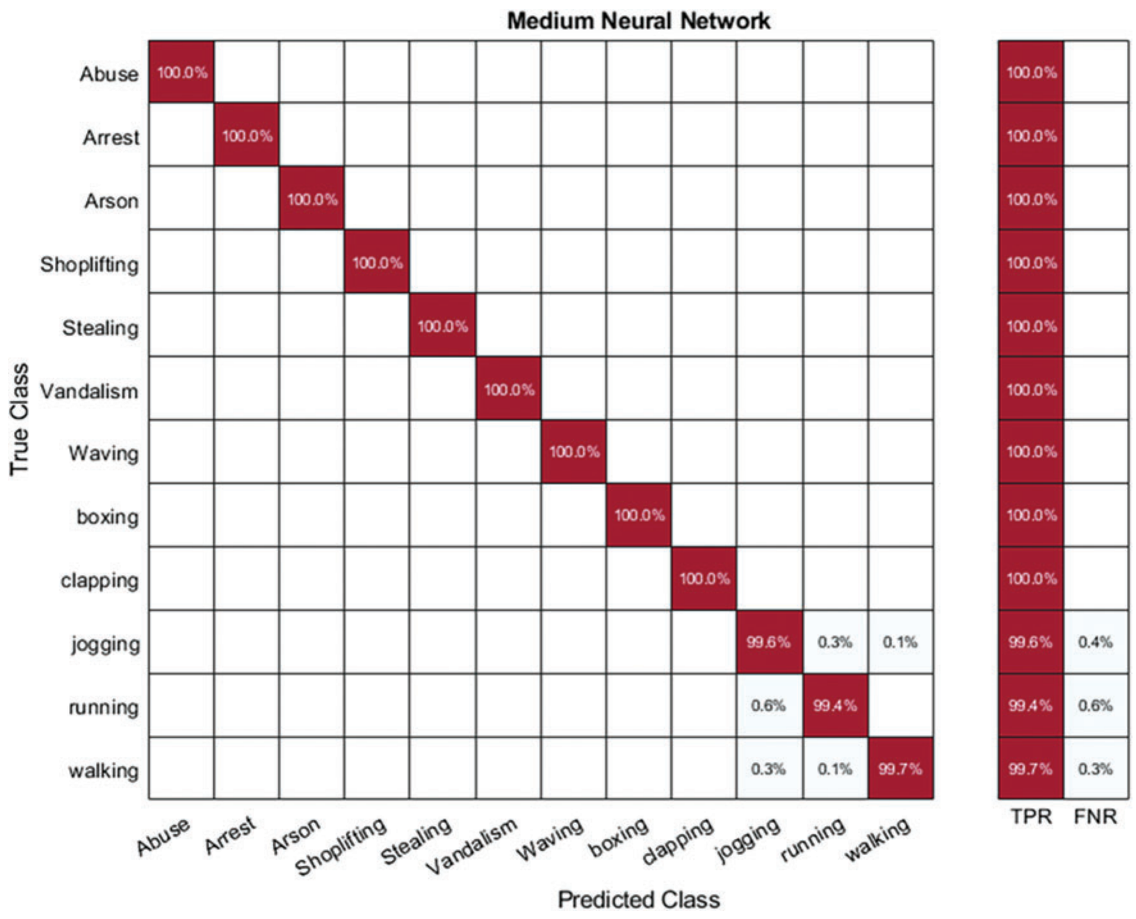

Figure 8: Class based confusion matrix of MNN classifier

Hybrid-KTH02 Dataset Results: In Table 5, the outcomes of the Hybrid-KTH02 dataset are presented. By analyzing Nasnet-Mobile deep features, the Wide Neural Network (WNN) classifier obtained the highest accuracy of 98.6%. The other classifiers also demonstrated high accuracy (>98%) along with Precision, F1 score, and Recall values shown in Table 5. The confusion matrix of the WNN classifier can be used to validate the model's performance. Fig. 10 shows the class-based confusion metric. Table 5 also provides the computation time for each classifier.

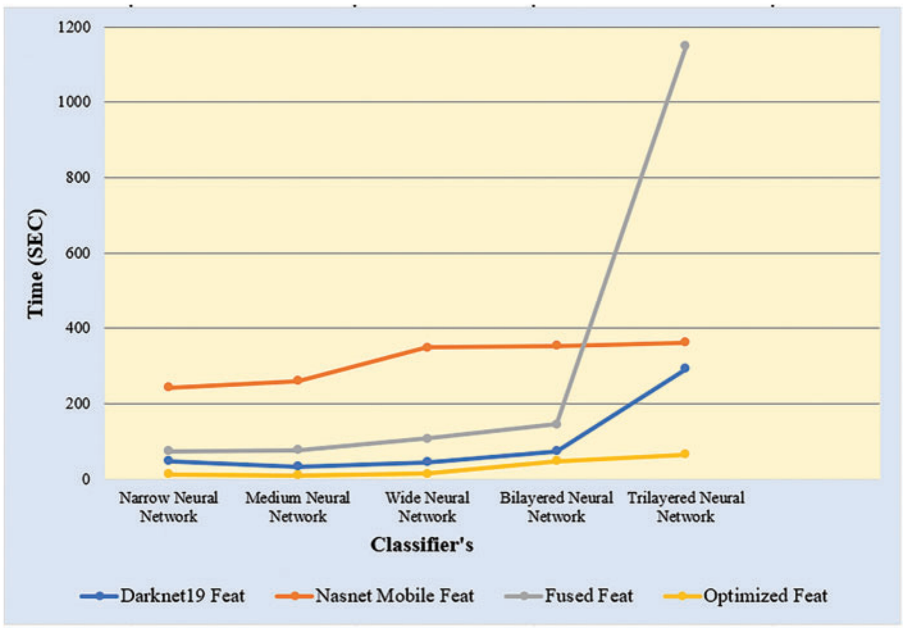

Figure 9: Time comparison of all neural classifiers on different features

Table 5: Classification results on Hybrid-KTH02 dataset using Nasnet Mobile deep features

Classifier	Parameters				
	Recall	Precision	F1 score	Accuracy (%)	Time (sec)
NNN	0.9825	0.9808	0.98	98.3	211
MNN	0.9816	0.98	0.9816	98.4	192
WNN	0.9862	0.9775	0.9825	98.6	302
BLNN	0.9837	0.9712	0.9775	98.2	400
TLNN	0.98	0.9783	0.9783	98	582

Hybrid-KTH02 Dataset Results: In Table 6, the outcomes of the Hybrid-KTH02 dataset are presented. By analyzing Darknet-19 deep features, the Wide Neural Network (WNN) classifier obtained the highest accuracy of 99.7%. The other classifiers also demonstrated high accuracy (>98%) along with Precision, F1 score, and Recall values shown in Table 6. The confusion matrix of the WNN classifier can be used to validate the model's performance. Fig. 11 shows the class-based confusion metric. Table 6 also provides the computation time for each classifier.

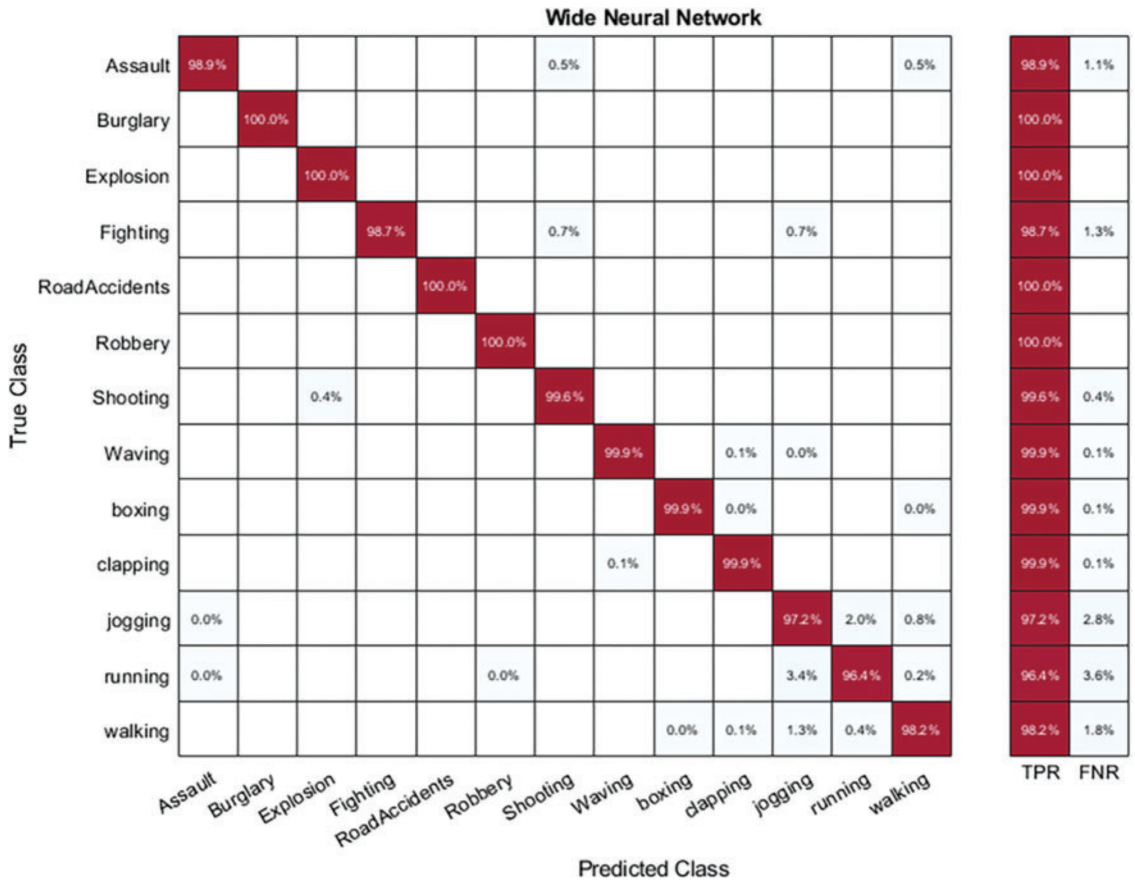

Figure 10: Class based confusion matrix of WNN classifier

Table 6: Classification results on Hybrid-KTH02 dataset using DarkNet19 deep features

Classifier	Parameters				
	Recall	Precision	F1 score	Accuracy (%)	Time (sec)
NNN	0.995	0.9966	0.995	99.6	38
MNN	0.9983	0.9983	0.9966	99.7	35
WNN	0.9983	0.9983	0.9966	99.7	50
BLNN	0.98	0.9783	0.9783	98.1	219
TLNN	0.984	0.9727	0.98	98.4	514

Hybrid-KTH02 Dataset Results: In Table 7, the outcomes of the Hybrid-KTH02 dataset are presented. By analyzing fused features, the Wide Neural Network (WNN) classifier obtained the highest accuracy of 99.7%. The other classifiers also demonstrated high accuracy (>99%) along with Precision, F1 score, and Recall values shown in Table 7. The confusion matrix of the WNN classifier can be used to validate the model's performance. Fig. 12 shows the class-based confusion metric. Table 7 also provides the computation time for each classifier.

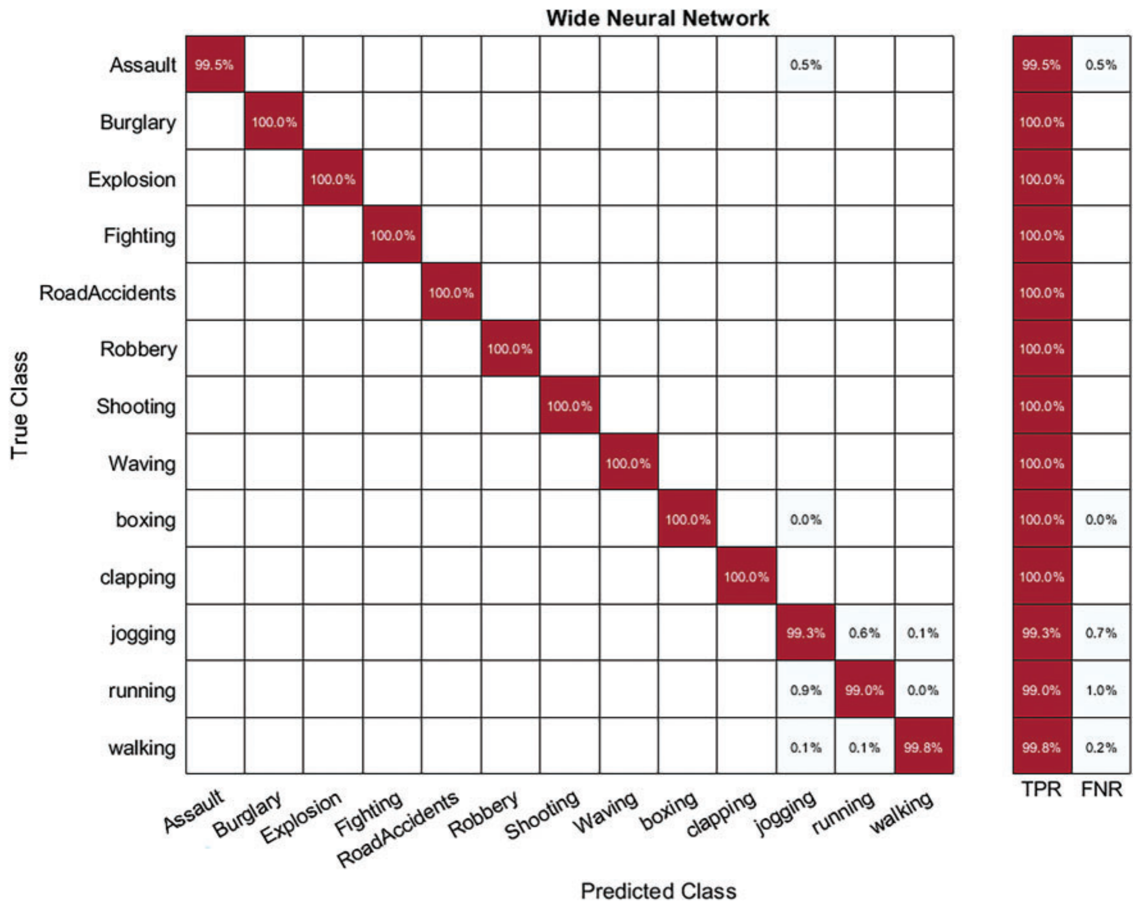

Figure 11: Class based confusion matrix of WNN classifier

Table 7: Classification results on Hybrid-KTH02 dataset using fused features

Classifier	Parameters				
	Recall	Precision	F1 score	Accuracy (%)	Time (sec)
NNN	0.9983	0.9983	0.9966	99.7	55
MNN	0.9983	0.9983	0.9966	99.7	27
WNN	0.9983	0.9983	0.9966	99.7	34
BLNN	0.9916	0.9961	0.9916	99.2	102
TLNN	0.9966	0.9958	0.9966	99.6	132

Hybrid-KTH02 Dataset Results: In Table 8, the outcomes of the Hybrid-KTH02 dataset are presented. By analyzing optimized features, the Wide Neural Network (WNN) classifier obtained the highest accuracy of 99.7%. The other classifiers also demonstrated high accuracy (>99%) along with Precision, F1 score, and Recall values shown in Table 8. The confusion matrix of the WNN classifier can be used to validate the model's performance. Fig. 13 shows the class-based confusion metric.

Fig. 13 shows the comparison of time of all NN classifiers. Fig. 14 provides the computation time for each classifier that shows the significance of feature optimization step.

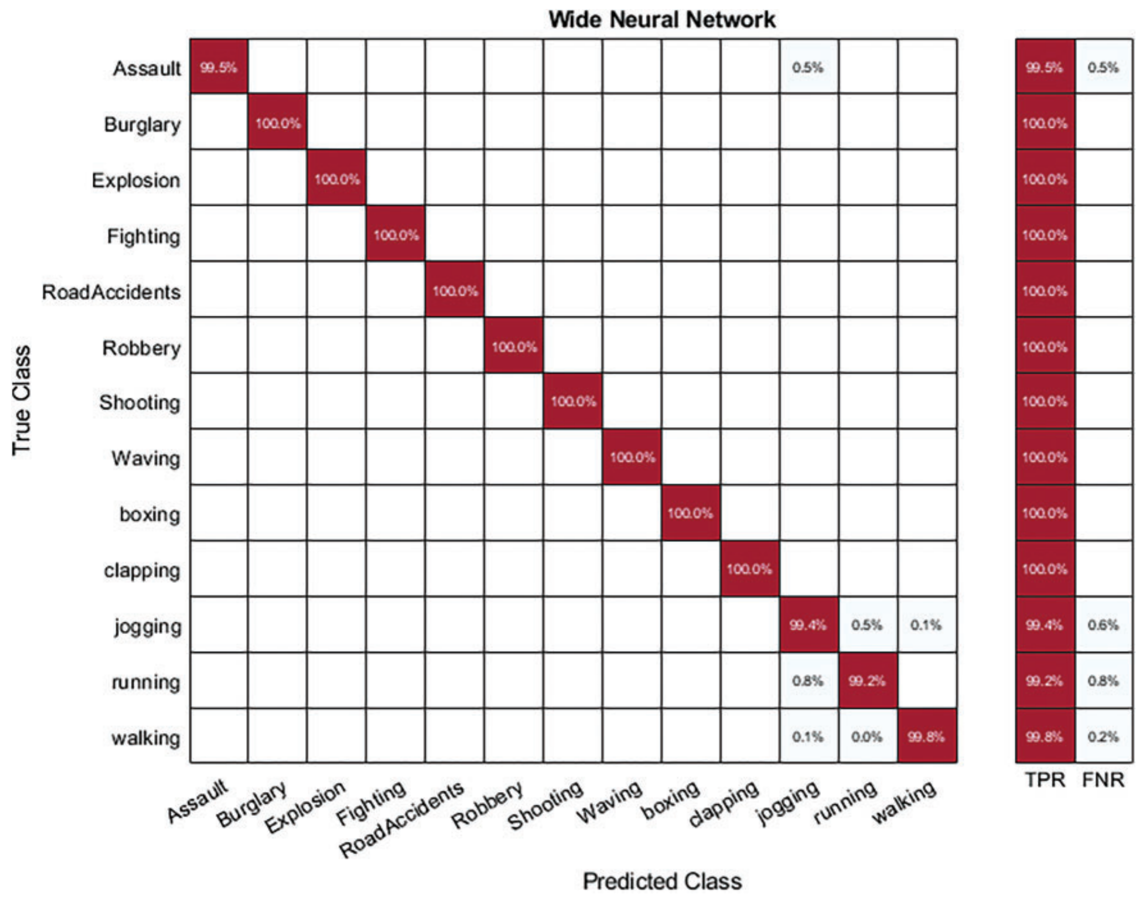

Figure 12: Class based confusion matrix of WNN classifier

Table 8: Classification results on Hybrid-KTH02 dataset using optimized features

Classifier	Parameters				
	Recall	Precision	F1 score	Accuracy (%)	Time (sec)
NNN	0.995	0.9966	0.995	99.6	14
MNN	0.9983	0.9983	0.9966	99.7	12
WNN	0.9983	0.9983	0.9966	99.7	15
BLNN	0.9883	0.9866	0.9866	98.8	45
TLNN	0.9887	0.9812	0.9837	98.7	89

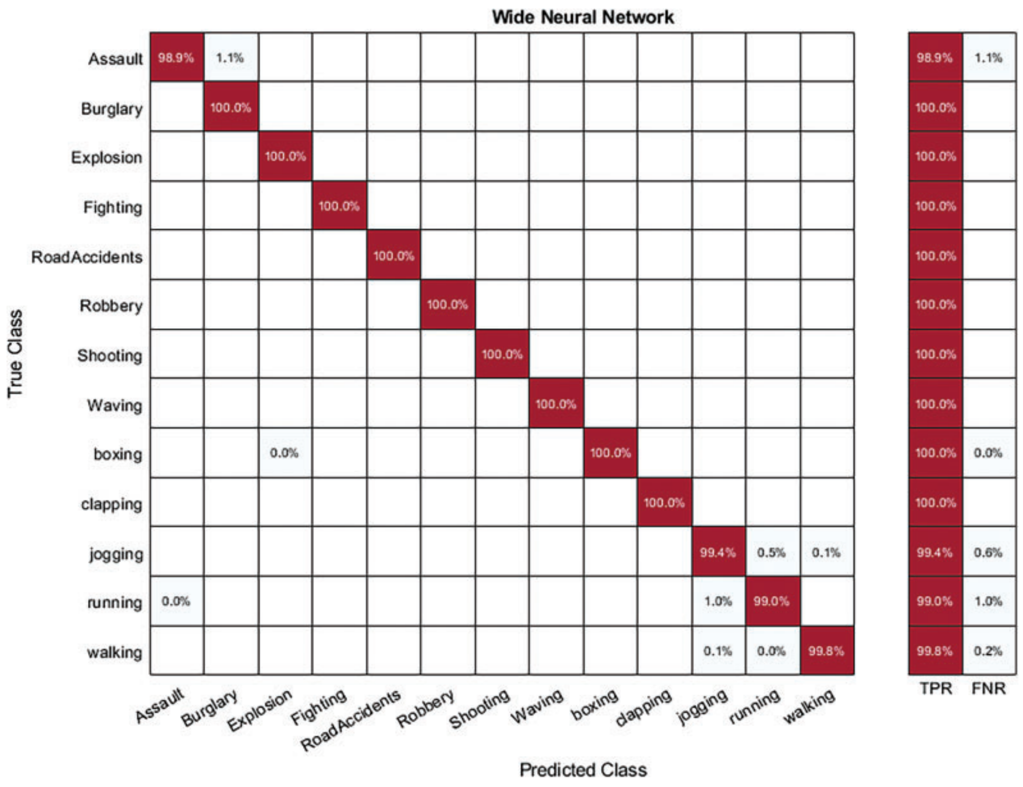

Figure 13: Class based confusion matrix of WNN classifier

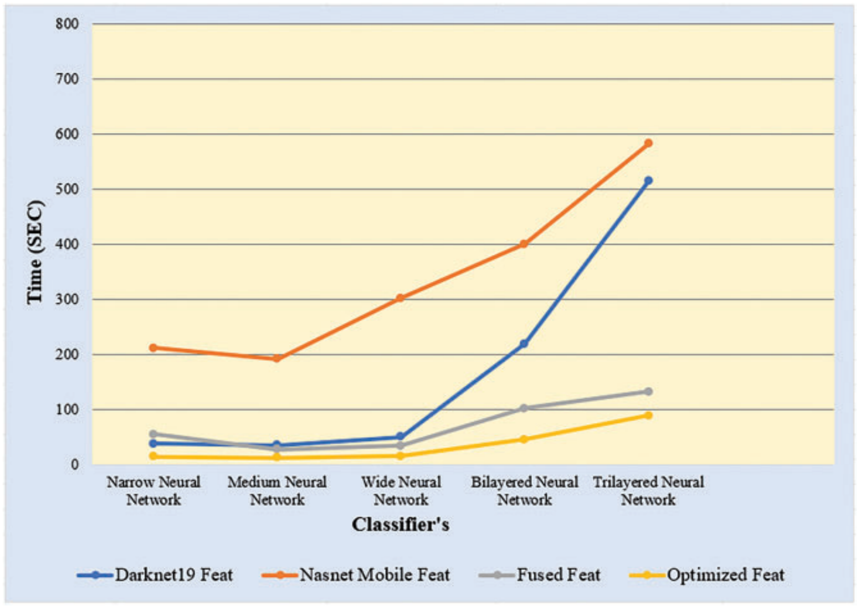

Figure 14: Time comparison of all neural classifiers on different features

4.3 *Comparison*

The results of comparing the accuracy of the proposed methodology with state-of-the-art (SOTA) methodologies using different datasets are shown in Table 9. The suggested framework achieved a higher accuracy of 99.8% and 99.7% on both action datasets compared to the current techniques. These numbers indicate that our AR framework outperforms SOTA approaches in terms of accuracy. In Table 10, we added a comparison among several neural nets such as VGG16, VGG19, ResNet50, GoogleNet, InceptinV3, DarkNet19, NasNet Mobile, and proposed framework. The each model is trained on the both selected datasets of this work and obtained the accuracy values for the comparison, as given below.

Table 9: Accuracy comparison of proposed method with different existing techniques

Reference	Year	Dataset	Accuracy (%)
[35]	2021	Public dataset	95.28
[36]	2020	CAVIAR dataset	87.15
[37]	2022	DCSASS dataset	98.38
Proposed	-	Hybrid-KTH01 dataset, Hybrid-KTH02 dataset	99.8, 99.7

Table 10: Comparison of selected deep models for the classification

Hybrid KTH01		Hybrid KTH02	
Model	Accuracy (%)	Model	Accuracy (%)
NasNet Mobile	**98.6**	**NasNet Mobile**	**98.6**
DarkNet-19	**99.7**	**DarkNet-19**	**99.7**
VGG16	98.4	VGG16	97.5
VGG19	97.6	VGG19	97.9
ResNet-50	98.4	ResNet-50	98.1
GoogleNet	95.2	GoogleNet	95.7
Inception-V3	96.9	Inception-V3	95.9

5 Conclusion

The applications of human action recognition (HAR) applications are gaining popularity in pattern recognition, computer vision, and machine learning, particularly for video surveillance purposes. A novel framework that employs deep learning and optimization algorithm is introduced in this article as a means of improving human action recognition. The study was carried out using an experimental process on two action datasets, namely Hbrid-KTH01 and Hybrid-KTH02. The proposed framework achieved recognition accuracy of 99.8% and 99.7% for each dataset. Comparison with state-of-the-art (SOTA) techniques revealed that the proposed framework outperformed recent techniques in terms of recognition accuracy. Furthermore, the fusion-based approach yielded better accuracy than individual deep learning features. The optimization algorithm played a significant role in reducing the execution time during the testing phase, and the optimization algorithm is used to reduce computational time without sacrificing classification accuracy. However, a few limitations are also noted in this work, such as reducing features using the proposed EO-based optimization dropped some important features that can mislead the visualization process. In the future, a more comprehensive deep model will be introduced using an efficient search-based optimization algorithm. In addition, large datasets like Muhavi, HMDB51, and UCF101 will be utilized for evaluation purposes.

Acknowledgement: This work was supported by the "Human Resources Program in Energy Technology" of the Korea Institute of Energy Technology Evaluation and Planning (KETEP) and granted financial resources from the Ministry of Trade, Industry, and Energy, Republic of Korea.

Funding Statement: This work was supported by the "Human Resources Program in Energy Technology" of the Korea Institute of Energy Technology Evaluation and Planning (KETEP) and granted financial resources from the Ministry of Trade, Industry, and Energy, Republic of Korea (No. 20204010600090).

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: S.K, M.A.K, and J.H.S; data collection: S.K, M.A.K, and J.H.S; draft manuscript preparation: S.K, M.A.K, and F.S; funding: T.K and J.C; validation: T.K, F.S and J.C; software: S.K, M.A.K, and F.S; visualization: J.C, T.K, and F.S; supervision: M.A.K and J.H.S. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The dataset used in this work is publically available for research purpose.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] G. Batchuluun, R. A. Naqvi, W. Kim and K. R. Park, "Body-movement-based human identification using convolutional neural network," *Expert Systems with Applications*, vol. 101, no. 6, pp. 56–77, 2018.

[2] R. K. Tripathi, A. S. Jalal and S. C. Agrawal, "Suspicious human activity recognition: A review," *Artificial Intelligence Review*, vol. 50, no. 1, pp. 283–339, 2018.

[3] N. Tasnim and J. H. Baek, "Dynamic edge convolutional neural network for skeleton-based human action recognition," *Sensors*, vol. 23, no. 3, pp. 778, 2023.

[4] G. Saleem, U. I. Bajwa and R. H. Raza, "Toward human activity recognition: A survey," *Neural Computing and Applications*, vol. 35, no. 2, pp. 4145–4182, 2023.

[5] J. K. Aggarwal and L. Xia, "Human activity recognition from 3D data: A review," *Pattern Recognition Letters*, vol. 48, no. 1, pp. 70–80, 2014.

[6] H. Arshad, W. Z. Khan, M. Alhaisoni and H. S. Hussein, "HGRBOL2: Human gait recognition for biometric application using Bayesian optimization and extreme learning machine," *Future Generation Computer Systems*, vol. 143, no. 3, pp. 337–348, 2023.

[7] F. Majeed, F. Z. Khan, M. Nazir, Z. Iqbal and M. Alhaisoni, "Investigating the efficiency of deep learning based security system in a real-time environment using YOLOv5," *Sustainable Energy Technologies and Assessments*, vol. 53, no. 2, pp. 102603, 2022.

[8] M. Ye, Q. Zhang, L. Wang, J. Zhu and R. Yang, "A survey on human motion analysis from depth data," in *Time-of-Flight and Depth Imaging, Sensors, Algorithms, and Applications: Dagstuhl 2012 Seminar on Time-of-Glight Imaging and GCPR 2013 Workshop on Imaging New Modalities*, NY, USA, pp. 149–187, 2013.

[9] F. Saleem, M. Alhaisoni, U. Tariq and A. Armghan, "Human gait recognition: A single stream optimal deep learning features fusion," *Sensors*, vol. 21, no. 4, pp. 7584, 2021.

[10] C. Tang, A. Tong, A. Zheng, H. Peng and W. Li, "Using a selective ensemble support vector machine to fuse multimodal features for human action recognition," *Computational Intelligence and Neuroscience*, vol. 22, no. 3, pp. 1–17, 2022.

[11] M. Nawaz, T. Nazir, A. Javed, U. Tariq and H. S. Yong, "An efficient deep learning approach to automatic glaucoma detection using optic disc and optic cup localization," *Sensors*, vol. 22, no. 1, pp. 434, 2022.

[12] X. Cheng, Y. Zhang, Y. Chen, Y. Wu and Y. Yue, "Pest identification via deep residual learning in complex background," *Computers and Electronics in Agriculture*, vol. 141, no. 5, pp. 351–356, 2017.

[13] X. Xia, C. Xu and B. Nan, "Inception-v3 for flower classification," in *2017 2nd Int. Conf. on Image, Vision and Computing (ICIVC)*, NY, USA, pp. 783–787, 2017.

[14] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 3, pp. 84–90, 2017.

[15] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, NY, USA, pp. 1251–1258, 2014.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[17] B. Koonce and B. Koonce, "EfficientNet," *Convolutional Neural Networks with Swift for Tensorflow: Image Recognition and Dataset Categorization*, vol. 11, no. 3, pp. 109–123, 2021.

[18] P. Tejaswi, S. Venkatramaphanikumar and K. V. K. Kishore, "Proctor net: An AI framework for suspicious activity detection in online proctored examinations," *Measurement*, vol. 206, no. 2, pp. 112266, 2023.

[19] T. Akram, M. Sharif, N. Muhammad and M. Y. Javed, "Improved strategy for human action recognition; experiencing a cascaded design," *IET Image Processing*, vol. 14, no. 5, pp. 818–829, 2020.

[20] N. u. R. Malik, S. A. R. Abu-Bakar, U. U. Sheikh, A. Channa and N. Popescu, "Cascading pose features with CNN-LSTM for multiview human action recognition," *Signals*, vol. 4, no. 2, pp. 40–55, 2023.

[21] H. Arshad, R. Damaševičius, A. Alqahtani and S. Alsubai, "Human gait analysis: A sequential framework of lightweight deep learning and improved moth-flame optimization algorithm," *Computational Intelligence and Neuroscience*, vol. 1, no. 5, pp. 223–253, 2022.

[22] I. M. Nasir, M. Raza, J. H. Shah, S. H. Wang and U. Tariq, "HAREDNet: A deep learning based architecture for autonomous video surveillance by recognizing human actions," *Computers and Electrical Engineering*, vol. 99, no. 6, pp. 107805, 2022.

[23] J. Jiang and Y. Zhang, "An improved action recognition network with temporal extraction and feature enhancement," *IEEE Access*, vol. 10, no. 2, pp. 13926–13935, 2022.

[24] G. Yang, Y. Yang, Z. Lu, J. Yang and D. Liu, "STA-TSN: Spatial-temporal attention temporal segment network for action recognition in video," *PLoS One*, vol. 17, no. 7, pp. e0265115, 2022.

[25] L. Shi, Y. Zhang, J. Cheng and H. Lu, "Action recognition via pose-based graph convolutional networks with intermediate dense supervision," *Pattern Recognition*, vol. 121, no. 1, pp. 108170, 2022.

[26] W. Yang, J. Zhang, J. Cai and Z. Xu, "HybridNet: Integrating GCN and CNN for skeleton-based action recognition," *Applied Intelligence*, vol. 53, no. 4, pp. 574–585, 2023.

[27] X. Shen and Y. Ding, "Human skeleton representation for 3D action recognition based on complex network coding and LSTM," *Journal of Visual Communication and Image Representation*, vol. 82, no. 3, pp. 103386, 2022.

[28] T. Liu, Y. Ma, W. Yang, W. Ji and R. Wang, "Spatial-temporal interaction learning based two-stream network for action recognition," *Information Sciences*, vol. 606, no. 7, pp. 864–876, 2022.

[29] F. Afza, M. Sharif, S. Kadry and G. Manogaran, "A framework of human action recognition using length control features fusion and weighted entropy-variances based feature selection," *Image and Vision Computing*, vol. 106, no. 1, pp. 104090, 2021.

[30] Y. Liu, H. Zhang, D. Xu and K. He, "Graph transformer network with temporal kernel attention for skeleton-based action recognition," *Knowledge-Based Systems*, vol. 240, no. 6, pp. 108146, 2022.

[31] F. Shehzad, M. Attique Khan, A. E. Yar, M. Sharif and M. Alhaisoni, "Two-stream deep learning architecture-based human action recognition," *Computers, Material and Continua*, vol. 74, no. 3, pp. 5931–5949, 2023.

[32] L. Zhang, C. P. Lim and Y. Yu, "Intelligent human action recognition using an ensemble model of evolving deep networks with swarm-based optimization," *Knowledge-Based Systems*, vol. 220, no. 11, pp. 106918, 2021.

[33] K. R. Mohan and G. Thirugnanam, "A dualistic sub-image histogram equalization based enhancement and segmentation techniques for medical images," in *2013 IEEE Second Int. Conf. on Image Information Processing (ICIIP-2013)*, NY, USA, pp. 566–569, 2013.

[34] A. Faramarzi, M. Heidarinejad, B. Stephens and S. Mirjalili, "Equilibrium optimizer: A novel optimization algorithm," *Knowledge-Based Systems*, vol. 191, no. 2, pp. 105190, 2020.

[35] A. J. S. Mehmood, "LightAnomalyNet: A lightweight framework for efficient abnormal behavior detection," *Computers, Material and Continua*, vol. 21, no. 13, pp. 8501, 2021.

[36] C. Amrutha, C. Jyotsna and J. Amudha, "Deep learning approach for suspicious activity detection from surveillance video," in *2020 2nd Int. Conf. on Innovative Mechanisms for Industry Applications (ICIMIA)*, NY, USA, pp. 335–339, 2020.

[37] E. Selvi, M. Adimoolam, G. Karthi, K. Thinakaran and N. M. Balamurugan, "Suspicious actions detection system using enhanced CNN and surveillance video," *Sensors*, vol. 11, no. 3, pp. 4210, 2022.