Advances in Science and Technology Research Journal

Advances in Science and Technology Research Journal, 2025, 19(9), 333–352
https://doi.org/10.12913/22998624/207850
ISSN 2299-8624, License CC-BY 4.0

Received: 2025.05.19
Accepted: 2025.07.15
Published: 2025.08.01

AI-driven cognitive surveillance framework for suspicious activity detection in academia

Asad Hameed Soomro[1,2*], Rafaqat Hussain Arain[1], Riaz Ahmed Shaikh[1]

[1] Institute of Computer Sciences, Shah Abdul Latif University, Khairpur Mirs, Sindh, Pakistan
[2] The Benazir Bhutto Shaheed University of Technology and Skill Development, Khairpur Mirs, Sindh, Pakistan
* Corresponding author's e-mail: asad.soomro31@yahoo.com

ABSTRACT

Early identification of human suspicious activities, especially in academia, is very crucial for enhancing security and safety, while existing surveillance systems remain ineffective at contextual behavior analysis. This study proposes the HYDPL Algorithm that incorporates a novel pseudo-labeling approach and innovative image analysis to perform human suspicious activity detection. The HYDPL algorithm includes several stages of data processing and feature extraction that aim to enhance the detection of activities by increasing the model's accuracy. To evaluate the effectiveness of the algorithm, two datasets were utilized: CampusWatch (Dataset-I), which consists of real-world scenarios specifically collected for this study from academia, targeting nine specific behaviors: kicking, punching, running, pushing, smoking, throwing, jumping, falling, and talking, providing realistic representation of human activities, and HMDB51, named as Dataset-II, which is a collection of movies or scripted videos. The evaluation results of the model were outstanding for both datasets, as the model was successful in delivering 98% accuracy for Dataset-I and 97% for Dataset-II, highlighting the model's ability to accurately detect real-world conditions. However, the study is limited to academia and only nine categories out of ten, with the tenth category being normal behavior. Future work will expand the dataset, explore advanced deep learning architectures, and implement real-time processing to enhance the model's applicability across various environments.

Keywords: computer vision, deep learning, human activity detection, academia, HYDPL, campuswatch, multimodal large models.

INTRODUCTION

Human activity recognition (HAR) has become a significant field of study, with precise identification of human actions generating considerable interest among researchers [1, 2]. This field seeks to determine a subject's behaviors using video recordings obtained by surveillance and/or smartphones cameras. Recordings are made as the individual engages in a set of specific actions, such as walking, laughing, nodding, driving, running, and punching. The information about people's behaviors is used by the researchers to meet demands from such domains like health care, fitness, or home automation [3], and to detect suspicious and non-suspicious activities in various environments.

Suspicious or abnormal behavior can vary depending on the context. In places like offices, airports, and banks, such behavior often includes actions like running, falling, jumping, fighting, or slipping. Recognizing these actions can help in identifying potential risks and ensuring safety in these settings [4]. In addition, examples may include illegal access to personal property, failure to pay a ticket at a subway station, and child abduction. If the place is an indoor facility such as a store, the unusual behavior becomes "shoplifting," "theft," or "burglary" [5]. Kicking, shoving, and punching can be reported as suspicious action to be noticed and identified in the video surveillance system.

Moreover, recent research [6] shows that the integration of AI with technologies such as computer vision, IoT, and edge computing significantly advances the capabilities of video surveillance across various environments. Considering this context, deep learning-based research is being conducted on the SAD in academic environment. In this study, an AI-based cognitive surveillance system, designed specifically for academic environments, is introduced. The system can analyze video input derived from several smartphone cameras, which is processed into individual frames to form the basis of analyzing anomalies. A fundamental component of our methodology is the Hybrid YOLO-DenseNet Pseudo Labeling (HYDPL) algorithm, which integrates state-of-art image processing techniques, interactive feature extraction with pretrained C3D encoders, and real-time object detection with YOLOv4-tiny model. To enhance the ability of analyzing the behaviors like running, falling and punching as suspicious the system employs the two-phase self-training scheme, resulting in more robust and accurate surveillance in academic environments.

Two datasets were utilized to assess the algorithm's performance: CampusWatch (Dataset-I), a collection of real-world scenarios, specifically collected for this study and the Human Motion Database 51 (HMDB51) [7], also referred to as Dataset-II, which is a collection of scripted videos or movie clips.

This article summarizes a research study on suspicious activity detection (SAD) in academia, focusing on: (1) gathering a new real-time HAR dataset from academia; and (2) implementation of a new multi-stage algorithm for HAR.

Related works

Recent progress in artificial intelligence and deep learning has resulted in the creation of multiple methods for the immediate observation and identification of potentially suspicious behaviors. Some of the existing research focuses on activity prediction in intelligent homes[8], detection of health emergencies[9], and abnormal actions in crowds [10]. Detection is also carried out in a variety of settings, such as shopping environments [11] and war field conditions [12]. As the area of activity detection evolves, new approaches have emerged to meet issues in a variety of settings and scenarios. The techniques discussed in this section are categorized into three main areas: AI-driven methods for HAR, YOLO-based applications for object detection, and the utilization of the HMDB51 dataset for action recognition in videos. Each category demonstrates unique approaches that contribute to the effectiveness of automated surveillance systems.

AI-driven techniques

Deep learning architecture has demonstrated remarkable versatility in both healthcare and surveillance applications. For instance, DenseNet121 has been successfully repurposed for pneumonia detection in chest X-rays [13] and COVID-19 diagnosis [14] underscoring its efficacy in medical imaging. Beyond healthcare, HAR systems increasingly leverage hybrid models to improve accuracy. [15] proposed a parallel CNN-transformer architecture to classify activities such as walking, sitting, and falling using wearable sensor data, while [16] employed raw accelerometer and gyroscope inputs with a 1D CNN model for real-time motion analysis. Multi-modal approaches are also gaining traction: [17] integrated CNN and ConvLSTM networks for smart healthcare applications, achieving robust performance through multi-level feature fusion. In surveillance contexts, [18] developed an optimized LSTM model using a Hybrid Spider Monkey-Chicken Swarm Optimization (HSM-CSO) technique to distinguish normal and abnormal behaviors at ATMs, highlighting the role of hybrid algorithms in edge-case scenarios.

YOLO applications for detection

In [19], an AI-based monitoring system using YOLOv8 was proposed for real-time crowd density evaluations and abnormal activity detection. The system excelled in detecting weapons or fires in crowded areas, thereby enhancing public safety through instant alerts and preventing crowd-smashing accidents. Building on the features of YOLO, [20] proposed YOLO-NL as a new object detector that enhances the identification of objects of various scales in intricate environments. The model utilized a global dynamic label assignment approach to balance precision detection and localization while improving CSPNET and PANET for detection. Further, FFCA-YOLO was developed as a lightweight detector specifically aimed at identifying small objects within remote sensing images [21]. This innovative approach incorporates lightweight elements designed to enhance features, facilitate fusion, and provide spatial context, making it effective for this specialized application.

HMDB51

A multi-stream architecture for HAR from video clips was proposed in [7], leveraging the Optical Flow Rhythm integration stream for temporal action recognition. The method was based on convolutional neural networks and weighted voting. The model was assessed against UCF101 and HMDB51 datasets and can pinpoint odd behavior. The HMDB51 dataset is one of the most valuable and practical resources for HAR model evaluation [22]. It contains about 7000 manually tagged video clips of human actions clustered into 51 classes.

DATASETS

Dataset-I

The CampusWatch dataset, referred to as Dataset-I and specifically collected for this research, aims to identify human suspicious behavior by analyzing video clips from various indoor and outdoor situations acquired by smartphone cameras in academic environments. This dataset provides approximately 23,041 frames categorized into ten classes: “kicking”, “punching”, “running”, “normal”, “pushing”, “smoking”, “throwing”, “jumping”, “falling”, and “talking”. (Figure 1) displays various activity frames from the collected dataset. The CampusWatch dataset used in this study is available at (https://rb.gy/bbdjlj).

The videos are recorded in SD and HD standards wherein the minimum frame rate is 30 fps which enables detailed behavior analysis. Each video clip ranges from 3 to 4 seconds, providing brief scenarios for recognizing suspicious activities. Dataset-I includes recordings from different educational institutions, presenting a variety of scenarios. The classes are balanced to ensure robust training and evaluation of models. Additionally, the dataset introduces challenges such as diverse camera angles, lighting variations, occlusions, and background clutter, which test the effectiveness of activity recognition algorithms.

Dataset-II

In addition, Dataset-II was also utilized for analysis. Derived from the HMDB51 dataset available on the Kaggle website [23], the HMDB51 dataset originally contains 51 classes. However, for this study, only 10 classes

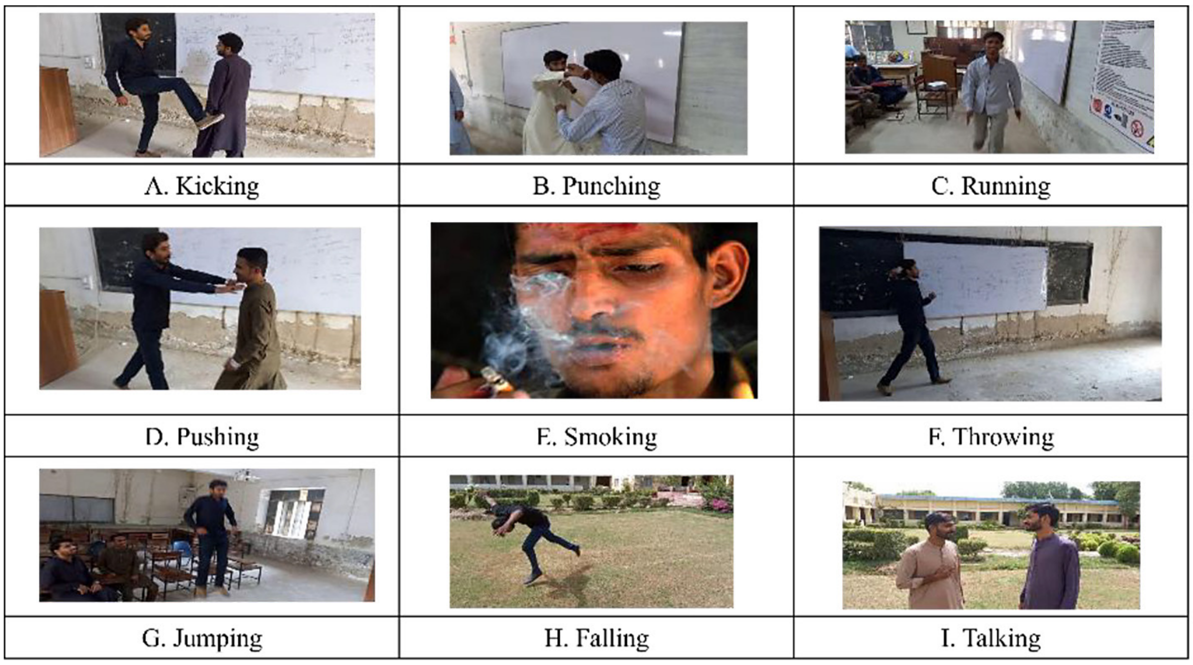

Figure 1. Example images for each activity in CampusWatch (Dataset - I)

relevant to suspicious activities were selected. Nine of these activities were classified as suspicious, while all other activities, except these nine, were categorized as normal. Furthermore, Information on the dataset, including the overall count of images (number of frames) corresponding to each of the ten activity types, is provided in (Table 1).

Data pre-processing

To ensure consistency and reproducibility, both datasets underwent the following preprocessing steps:

Frame downsampling

Videos in Dataset-I (CampusWatch) were recorded at 30 fps. To balance computational efficiency and temporal resolution, we extracted 5 frames per second (fps) using motion-based keyframe selection [see Section Frame extraction]. This reduced redundancy while retaining critical action phases.

Frame resizing and standardization

All frames are resized to 224 × 224 pixels using bilinear interpolation (Figure 4).

Ethical constraints

CampusWatch data was anonymized, and participants provided written informed consent via a dedicated consent form, adhering to ethical guidelines for surveillance research.

METHODS

A multi-stage approach-based algorithm, Hybrid YOLO-DenseNet Pseudo Labeling (HYDPL), is designed and implemented in this study for robust detection of suspicious activities. To conduct the experiments, the academic environment is considered. The developed algorithm integrates state-of-the-art methods for analyzing images, extracting important features, and detecting objects. The research plan commenced with the acquisition of videos through multiple smartphone cameras, followed by the transformation of video streams into discrete frames using visual consistency measure and reconstruction quality measure, as adopted by [24]. These frames undergo preprocessing to standardize image sizes and enhance visual features, which are critical for the subsequent analysis, as discussed in [25]. Henceforth, the YOLOv4-tiny algorithm, selected for its real-time performance and precision, is employed for object detection as described in [26], while DenseNet121 [27] with Global Average Pooling [28] is used to extract and refine features. The architecture of this integrated approach is illustrated in (Figure 2), providing a comprehensive overview of the model's workflow. Henceforth, the refined data feeds into a two-stage self-training scheme, enabling the iterative improvement of pseudo-labels and anomaly detection accuracy. Finally, the classification layer [29] employs the enhanced feature vector along with the video-level label Y to assess whether the activities being observed are considered suspicious or not. The research process is illustrated in (Figure 3) and the detailed steps of the HYDPL are described in Algorithm 1.

Table 1. Overview of activity categories and frame counts

S. No	Category	Dataset-I	Dataset-II	Sub Total
1	Kicking	2.744	81,862	84606
2	Punching	2.150	130,162	132312
3	Running	2.110	78,930	81040
4	Normal	2.700	82,811	85511
5	Pushing	1587	157,312	158899
6	Smoking	2.090	63,589	65679
7	Throwing	1.950	92,300	94250
8	Jumping	2.700	162,930	165630
9	Colliding	2.300	95,162	97462
10	Talking	2.710	155,282	157992
Total =		23,041	1,100,340	1,123,381

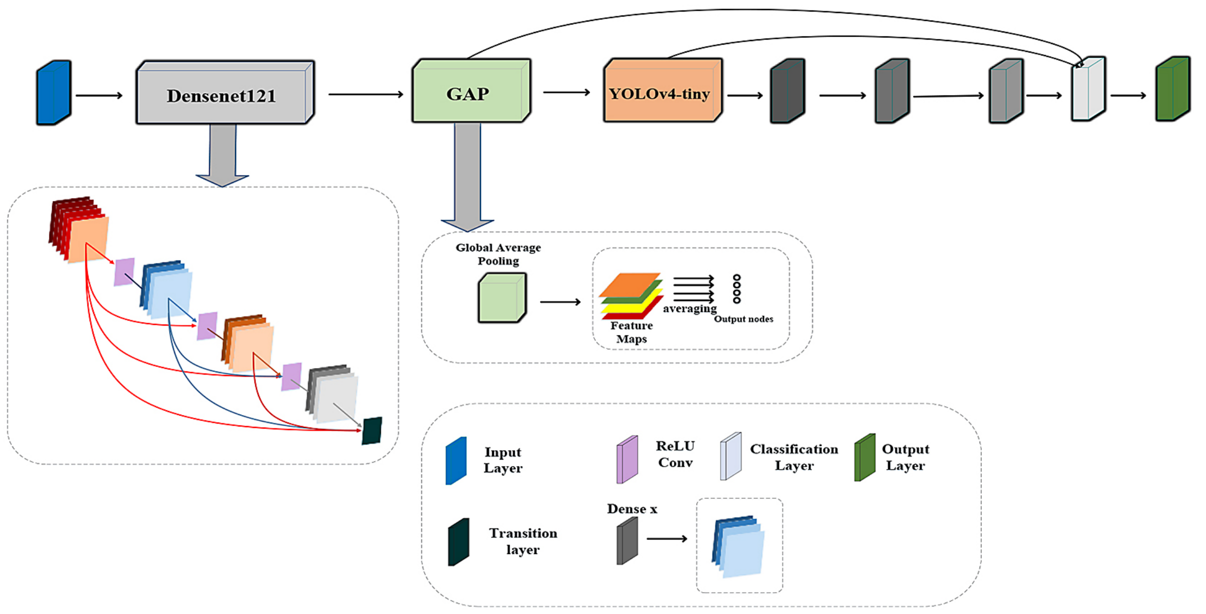

Figure 2. Framework for activity recognition model

Algorithm 1. Hybrid YOLO-DenseNet Pseudo Labelling (HYDPL)

```
Initialize
Define video set $V$ from the input where each video $V_j \in V_1, V_2, \ldots\ldots, V_N$
For each video $V_j$:
    Convert video $V_j$ into a series of frames $F_j = f_{1j}, f_{2j}, \ldots \ldots, f_{nj}$
    Apply preprocessing $C$ on frames $F_j$ :
                Standardize image sizes using OpenCV resizing.
For each frame $f_{ij} \in f_j$ :
    Detect objects using YOLOv4-tiny.
For each frame $f_{ij}$ with detected objects:
    Extract features $D_{ij}$ using DenseNet121
    Apply Global Average Pooling (GAP) on extracted features.
Generate pseudo-labels $L_{ij}$ for each frame based on initial detection results.
Employ a two-stage self-training process:
    Stage 1: Train the model with initial pseudo-labels.
    Stage 2: Refine labels and retrain the model iteratively.
Identify suspicious activities A in each frame based on the refined pseudo-labels and features.
Estimate anomaly scores for each video clip based on refined pseudo-labels.
Classify videos into suspicious activities based on detected behaviours.
Validate the performance on datasets:
    Dataset-I: Real-world academic environment scenarios.
    Dataset-II: HMDB51, staged video clips.
Generate final detection report $R_j$ for each video $V_j$.
If validation performance is unsatisfactory, revisit steps 3-11.
    End
```

Frame extraction

Frame extraction is a critical component of the HYDPL data preparation process. Videos were split into individual frames using OpenCV function [30]. To ensure representative and non-redundant keyframes, we employed continuity, priority, and repetition avoidance criteria [see Section Image resizing and standardization].

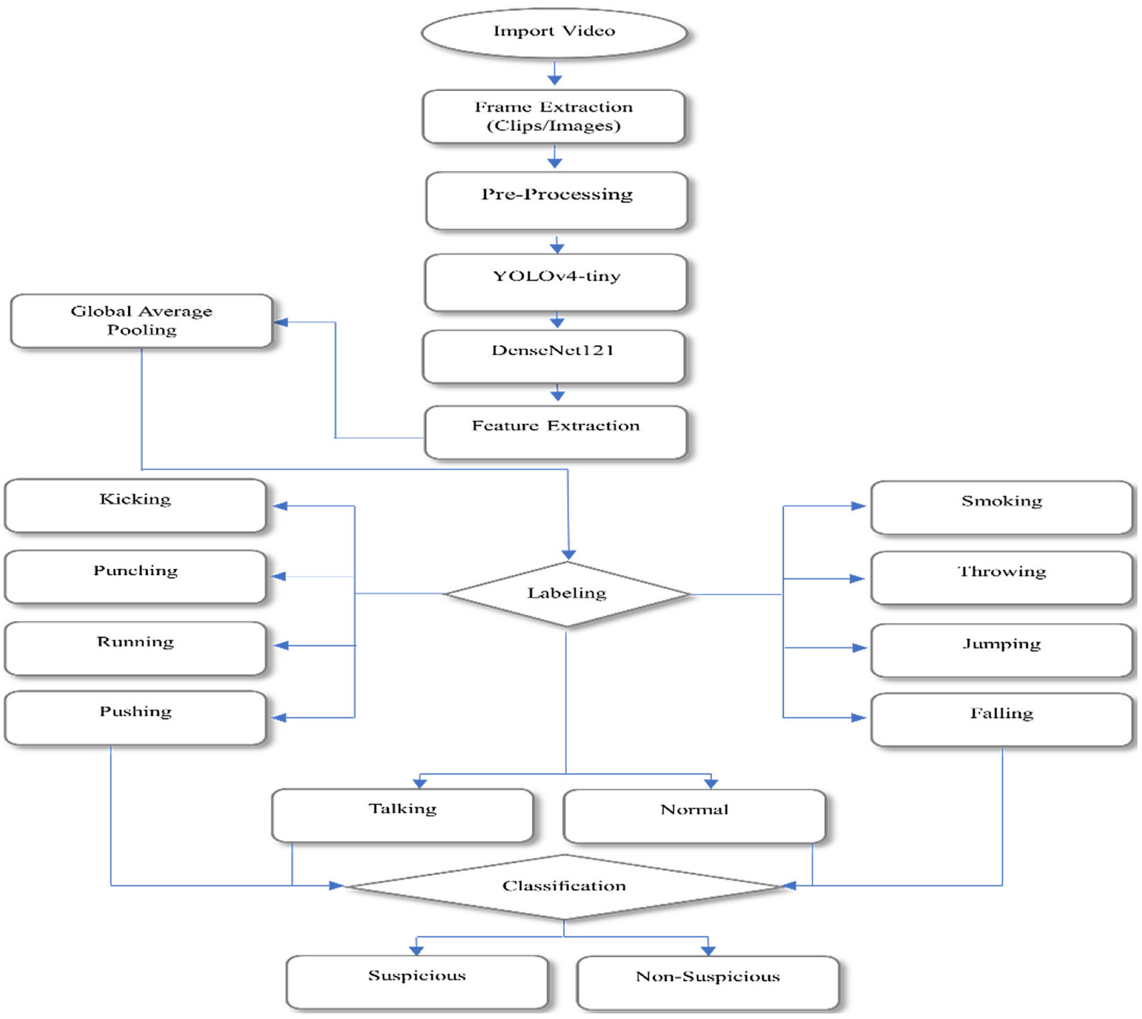

Figure 3. Activity detection methodology

The visual consistency measure (VCM) and reconstruction quality measure (RQM) were used to validate frame quality. VCM evaluates preservation of significant actions, while RQM assesses reconstruction fidelity of the original video sequence [see Section Image resizing and standardization].

Pre-processing (image resizing)

As detailed in [see Section Data Pre-processing], all frames are resized to 224 × 224 pixels (Figure 4) using bilinear the OpenCV library [30] facilitated this standardization, ensuring consistent input dimensions for the model.

YOLOv4-tiny for object detection

YOLOv4-tiny is applied to enhance object recognition from real-time video frames. This lightweight model is based on CSPDarknet53-tiny, which improves model learning by splitting and combining feature maps using CSPBlock modules, demonstrating high efficiency with low computational cost. Parallelly, the LeakyReLU activation function helps in faster and more effective feature extraction, and the Feature Pyramid Network helps in faster multi-scale feature extraction. Henceforth, the prediction process is based on grid-based image analysis accompanied by the bounding boxes, whereby the Jaccard Overlap score, loss function aims at fitting the vital detection and classification.

DenseNet121

In our study, DenseNet121 is employed as the backbone for feature extraction, utilizing its dense connectivity and efficient parameter usage to enhance the model's ability to capture robust features, crucial for accurate anomaly detection.

Figure 4. Image A (on the left) displays the original frame, while Image B (on the right) illustrates the same frame resized to 224 × 224 pixels using OpenCV

The architecture comprises multiple dense blocks, with each layer receiving input from all preceding layers, maximizing information flow and mitigating the vanishing gradient problem. To improve performance, transition layers are added between these dense blocks, minimizing the size of feature maps and increasing computing efficiency. These layers include batch normalization for stable training, 1 × 1 convolution to reduce the number of feature maps, and 2 × 2 average pooling to downsample spatial dimensions, allowing for efficient processing of high-dimensional data while preserving key features for accurate classification and anomaly detection.

Backbone selection rationale

DenseNet-121 was selected as the backbone due to its superior feature reuse, gradient flow, and parameter efficiency [31]. Comparative studies in activity recognition and video anomaly detection confirm its higher accuracy and stability over other CNN architectures like ResNet50 and VGG architectures [32, 33], making it ideal for surveillance tasks.

Global average pooling for feature extraction

After frames extraction and evaluation of the visual consistency and the quality of the reconstructed image, the features are obtained with the help of global average pooling (GAP). This procedure is used to further reduce the dimensionality of the spatial domain of the extracted frames while maintaining attributes that would be beneficial in detecting suspicious activity.

Feature extraction and labeling

Identifying and classifying suspicious activities in video footage relies heavily on feature extraction and labeling. Each frame taken from a video is treated as an individual instance within a framework known as the bag-of-instances (BoI) model. These instances are combined to create a refined feature vector that encapsulates the key characteristics of the video content. In this context, the label $Y \in \{1,0\}$ indicates whether suspicious activity is present or absent in the video. The extracted feature vector is then optimized or further adjusted to suit the classification layer. Such labeling of critical features at this stage affects the model in a precise manner, enabling it to distinguish between normal and suspicious activities, thereby boosting the chances of a lasting classification.

Classification layer

The final step towards activity detection is to identify the observed activity as suspicious or non-suspicious. After condensing the feature maps by GAP, the fine-tuned feature vector is arranged along with the pseudo-labels obtained in the Feature Extraction and Labeling step, which assist the classification process. These inputs, including the video-level label, are passed into the classification layer, which helps the final classification decision. This layer uses a fully connected layer and applies a SoftMax activation function to the logits obtained and provides probability scores in the two classes namely either suspicious or non-suspicious.

EXPERIMENTS

Frame extraction

The process of frame extraction begins with maintaining a steady frame rate, denoted as *f*, This frame rate can be determined using the formula:

$$f = \frac{N_{total}}{T} \tag{1}$$

where: *T* represents the total duration of the video in seconds, while N_{total} refers to the overall number of frames present in the video. Priority is managed by assigning higher weights to frames depicting significant actions, defined by the priority function *P(t)* as shown in Equation 2.

$$\mathrm{P(t)} = \begin{cases} \alpha \text{ if frame at time t contains A} \\ \quad 1 \text{ otherwise} \end{cases} \tag{2}$$

where: α represent a constant greater than 1, which serves to enhance the likelihood of detecting frames that exhibit suspicious activities.

Repetition avoidance ensures non-redundant frame extraction to reduce the computational cost and more accurate detection. The relationship between frames *i* and *j* is expressed through a similarity measure, denoted as *S (i, j)*, which is defined in Equation 3.

$$S\,(i,j\,) = \frac{1}{n}\sum_{k=1}^{n}\left|F_i[k] - F_j[k]\right| \tag{3}$$

Frames are considered redundant if *S (i, j)* exceeds a threshold *θ*. This combination of continuity, priority, and repetition avoidance ensures that the selected frames are informative and non-redundant, optimizing the data for further analysis.

Visual consistency measure

The video comparison metric (VCM) is determined by employing the semi-Hausdorff distance, which serves as a method to evaluate and contrast the original video sequence *V*. This approach is outlined in Equation 4:

$$V = \{F_v\,(t+n)|n = 0,1,\dots.\delta NF\,\}\cdot \tag{4}$$

with the extracted key frames *K*, expressed in Equation 5:

$$K = \{F_k\,(t+n_k)|n_k \epsilon\, I_k\,\} \tag{5}$$

To quantify the distance between a frame F_v *(t + n)* in the video sequence *V* and the set of key frames *K*, we compute as written in Equation 6:

$$\delta = (F_v\,(t+n), K) = min_j \\ \{Diff(F_v\,(t+n), F_k\,(t+n))|j = 1,2,\dots,\xi NKF\} \tag{6}$$

where: *Diff(.)* is a suitable frame difference measure, such as pixel-wise difference or structural similarity index.

The maximum frame distance Δ*(V, K)* between the video sequence *V* and the set of key frames *K* is expressed in Equation 7:

$$\Delta(V,K) = max_n \\ \{\delta(F_v\,(t+n),K)|n = 0,1,\dots,\xi NF\; -1\} \tag{7}$$

Finally, Visual Consistency Measure *F(V, K)* can then be computed using Equation 8:

$$\mathcal{F}(V,K) = MaxDiff - \Delta(V,K) \tag{8}$$

Higher values of *F(V, K)* indicate that the extracted frames effectively represent the video, ensuring that the model receives frames with crucial information for accurate classification.

Reconstruction quality measure (RQM)

RQM uses frame interpolation FIA(.) to generate interpolated frames $\tilde{F}\,(t+n)$ from the key frames *K*, defined in Equation 9:

$$\tilde{F}\,(t+n) - FIA \\ (t+n_k), F_k\,(t+n_{j+1}), n, n_j, n_{j+1}) \tag{9}$$

The RQM is calculated by comparing the original frames F_v *(t + n)* with the interpolated frames $\tilde{F}\,(t+n)$ using a Frame Similarity Measure (FSM) as expressed in Equation 10:

$$RQM\,(V,K) = \\ = \sum_{n=0}^{N_{V-1}} FSM\left(F_v\,(t+n), \tilde{F}\,(t+n)\right) \tag{10}$$

The FSM is modeled after a PSNR-like measure, expressed in Equation 11:

$$FSM\left(F_v\,(t+n), \tilde{F}\,(t+n)\right) = \\ = Clog\left(\frac{\mathrm{MaxDiff}}{\mathrm{Diff}\left(F_v\,(t+n), \tilde{F}\,(t+n)\right)}\right) \tag{11}$$

where: C is a constant, and $Diff(.)$ is the frame difference distance. The motion-based down sampling (30 fps → 5 fps) ensured non-redundant keyframes while maintaining an RQM score of 89.7%, confirming reconstruction fidelity. This measure ensures that the key frames allow for accurate reconstruction, preserving the sequence integrity necessary for detecting suspicious activities.

Image resizing and standardization

The resizing and normalization procedures retrieved key frames based on continuity, priority, and non-redundancy approaches. Therefore, maintaining the integrity of video content relies on these key frames, which contain the most informative and non-redundant data. For consistency, each extracted frame was processed using the "resize_and_save_image" function.The function steps are defined in Algorithm 2.

To achieve consistent and accurate model input, the extracted frames were uniformly scaled, as depicted in (Figure 4) Further, the implementation details are discussed in [25].

YOLOv4-tiny for object detection

The standardized 224 × 224 pixel images provide a consistent input format, ensuring that the YOLOv4-tiny model can effectively detect objects within each frame while maintaining high accuracy.

Network structure

The base network, CSPDarknet53-tiny [26], utilizes a CSPBlock module, enhancing gradient flow and learning capabilities compared to earlier YOLO versions, although it increases computation by 10–20%. To improve efficiency, we employed the LeakyReLU activation function, defined as shown in Equation 12:

Algorithm 2: Image resizing

```
Initialize
    Receive two arguments:
    path_to_input: This is the location of the input image file.
    path_to_output: This is the location where you want to save the resized image.
    Utilize the cv.imread function to read the image from the input_path. Store the result in a variable
    named image.
    Use the cv.resize function to adjust the image size to 224 × 224 pixels. The resized image should
    be saved in a variable called resized_image.
    Store the resized image at the given output location using cv.imwrite.
    Display a message indicating that the image resizing and saving process was successful
    End
```

$$y_i = \begin{cases} x_i \; if \; x_i \; \geq 0 \\ \dfrac{x_i}{a_i} \; if \; x_i \; \leq 0 \end{cases} \qquad (12)$$

where: $x_i \in (1, \infty)$ and a_i is a constant parameter.

Feature fusion is achieved through a feature pyramid network (FPN) that processes feature maps of sizes 13 × 13 and 26 × 26 optimizing speed while maintaining detection accuracy.

YOLOv4-tiny prediction process

The prediction process begins by resizing the input images and segmenting them into a grid of S × S cells.YOLOv4-tiny is designed to redict several bounding boxes within each cell. To evaluate the accuracy of these predictions, a metric known as the "Jaccard Overlap" score $Jacc_{(i,j)}$ is computed using Equation 13:

$$\text{Jacc}_{(i,j)} = P_{(i,j)} \times \text{Jaccard}_{(pred \cap truth)} \qquad (13)$$

where: $P_{(i,j)}$ indicates whether an object is present in the cell, and $\text{Jaccard}_{(\text{pred} \cap \text{truth})}$ measures the overlap between the predicted bounding box and the actual ground truth box, serving as an indicator of how well the prediction aligns with the true location of the object.This approach helps assess both the overlap and the accuracy of localization of objects within the predicted bounding boxes.

A threshold is applied based on the Jaccard Overlap score to streamline predictions and reduce redundancy. Here only bounding boxes with Jaccard Overlap scores greater than threshold value are preserved for further processing. Further, (Figure 5) shows the overall prediction schema in YOLOv4-tiny.

YOLOv4-tiny loss function

To enhance the model's effectiveness during training, YOLOv4-tiny utilizes a detailed loss function that consists of three primary components:

- *Detection Loss ($loss_{detect}$):* This aspect of the loss function penalizes both incorrect object detections and errors in localization, relying on the Jaccard Overlap scores as outlined in Equation. 14.

$$\text{loss}_{\text{detect}} = -$$

$$= -\sum_{i=0}^{S^2}\sum_{j=0}^{B} W_{ij}^{obj}\left[\hat{C}_i^j \log\left(\hat{C}_i^j\right) + \left(1-\hat{C}_i^j\right)\log\left(1-\hat{C}_i^j\right)\right] \quad (14)$$

- *Category Loss ($loss_{cat}$):* This component assesses the inaccuracies in predicting object categories.

Bounding box regression loss *($loss_{bbox}$)*: This part of the loss function evaluates how accurately the predicted bounding box coordinates align with the actual ground truth, optimizing the spatial localization.

DenseNet121

The input frames denoted as $F = \{f_i\}_{i=1}^{N}$ include the regions identified by YOLOv4-tiny. DenseNet121 processes these frames to generate comprehensive feature maps denoted as $\bar{F}$. The architecture's dense connectivity, expressed mathematically in Equation 15, ensures robust feature propagation:

$$x_\ell = H_\ell([x_0, x_1, \dots\dots, x_{\ell-1}]) \quad (15)$$

where: x_ℓ is the output of the ℓ-th layer, H_ℓ (.) is a function that comprises of batch normalization, ReLU and convolution operations and $[x_0, x_1, \dots\dots, x_{\ell-1}]$ is the vector that combines feature maps of all preceding layers.

Transition layers

To manage the dimensionality and complexity of these feature maps, transition layers were integrated between the dense blocks within DenseNet121. These transition layers were essential for optimizing the model's efficiency in processing input frames. After each dense block, a transition layer was applied to down-sample the feature maps.

The transition process started with Batch Normalization to stabilize the training by normalizing the activations inside each mini batch. Then, a 1 × 1 convolution was performed to minimize the number of feature maps. The output of this convolution, x_{conv}, was calculated using Equation 16:

$$x_{conv} = w_{conv} \times x_{bn} + b_{conv} \quad (16)$$

where: X_{bn} refers to the input that has undergone batch normalization, while w_{conv} denotes the weights associated with the convolutional layer, and b_{conv} represents the bias term. This process effectively compresses the feature maps, striking a balance

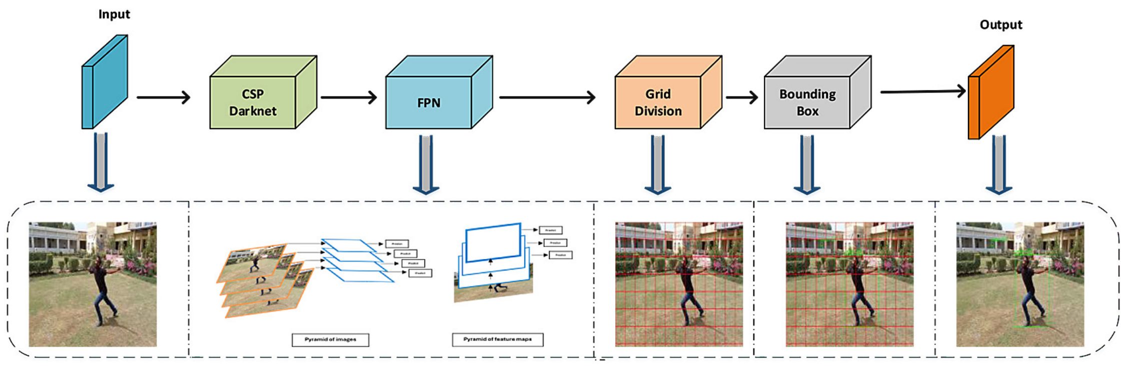

Figure 5. YOLOv4-tiny prediction process

between preserving essential information and minimizing the computational load.

Finally, a 2 × 2 average pooling layer was utilized to down-sample the feature maps and reduce the spatial dimension of the feature maps to half. This pooling operation is mathematically described in Equation 17:

$$x_{pool}\,(i,j) = = \frac{1}{4}\sum_{m=0}^{1}\sum_{n=0}^{1} x_{conv}\,(2i + m, 2j + n) \tag{17}$$

These feature maps were then transferred to the Global Average Pooling (GAP) layer. The GAP layer transformed these spatially reduced maps into a feature vector that retained essential spatial information, making it highly effective for the classification and anomaly detection tasks in our study.

Global average pooling operation

The key video frames are represented as as $X \in \mathbb{R}^{C \times H \times W}$ where (C) is the number of channels, (H) is the height, and (W) is the width of the frames. The pooling function f_{AP} calculates the output feature map (Y) by averaging the pixel values within a specified window. This average pooling function is mathematically defined in Equation 18:

$$Y\, c', i', j' = = \frac{1}{k^2}\sum_{u=0}^{k-1}\sum_{v=0}^{k-1} X_{c, s\times i'+u, s\times j'+v} \tag{18}$$

where: c' represents the channel index of the output feature map Y, the term (c') refers to the index of the channel in the output feature map Y. The variables i' *and* j' indicate the spatial locations within this output feature map (Y). The expression $X_{c,sxi'u,sxj'+v}$ represents the pixel value located at the position $s \times i' + u$, $s \times j' + v$ in the input frame X. The output feature map Y has spatial dimensions H' and W′, which are determined as outlined in Equation 19 and 20:

$$H' = \left\lfloor \frac{H-k}{s} \right\rfloor + 1 \tag{19}$$

$$W' = \left\lfloor \frac{W-k}{s} \right\rfloor + 1 \tag{20}$$

The down-sampled feature maps are obtained by down-sampling the original feature maps, which retain only the spatial information for accurate human activity identification. It also minimizes the model's complexity, allowing it to focus on relevant data, for processing large volumes of real-time information.

Feature extraction and labeling

This phase is pivotal for effectively identifying and classifying suspicious activities within the video sequences.

Frame representation and video-level labeling

Consider a series of extracted frames $F = \{f_i\}_{i=1}^{N}$ from a video, where each frame f_i represents an instance. The video-level label Y indicates the presence or absence of anomalies, following the BoI representation.

Feature extraction with pretrained encoders

The frames, denoted as F, underwent processing through a pretrained feature encoder like C3D. This method was utilized to extract features from each frame, resulting in collections of features represented as $\bar{F}$. Given a pair of bags (e.g., a positive bag B^a with anomalous frames and a negative bag B^n with normal frames), we employ a self-training approach to estimate anomaly scores for each frame. Let $\{s_i^a\}_{i=1}^{N}$ and $\{s_i^n\}_{i=1}^{N}$ denote the anomaly scores for frames in B^a and B^n, respectively.

Pseudo label generation and feature encoder fine-tuning

The clip-level pseudo labels $\hat{Y}^a = \{\hat{Y}_i^a\}_{i=1}^{N}$ are generated from estimated anomaly scores through a smoothing function $\hat{Y}_i^a$ = smoothing_fumction (s_i^a). which normalizes and smooths these scores. A pseudo label generator G is trained with a deep Multiple Instance Learning (MIL) ranking loss, aiming to optimize G parameters to produce accurate clip-level pseudo labels, as depicted in (Figure 6, Pseudocode).

Feature encoder architecture

Figure 7 illustrates the architecture of our proposed feature encoder ESG_A, derived from a vanilla feature encoder (e.g., C3D) enhanced with a self-guided attention mechanism. The encoder is optimized using the estimated pseudo labels,

Pseudocode. Pseudo label generation and encoder fine-tuning

```
Initialize
Input: Anomalous frames bag $B^a = \{f_1,\dots,f_N\}$
    | Pretrained feature encoder E (e.g., C3D)
    | Initial anomaly scores s = $\{s_1,\dots,s_N\}$
Step 1: Feature Extraction
    | $\overline{F} = E(B^a)$
Step 2: Estimate Anomaly Scores (already computed)
    | $s_i^a$ = anomaly score for frame i
Step 3: Generate Pseudo Labels
    | for i in range(N):
    | $\hat{Y}$[i] = sigmoid($s[i]$)
Step 4: Fine-tune Encoder ESGA using pseudo labels
    | loss = MIL_Ranking_Loss(ESGA($\overline{F}$), $\hat{Y}$)
    | Backpropagate(loss) and update θ_ESGA
    | End
```

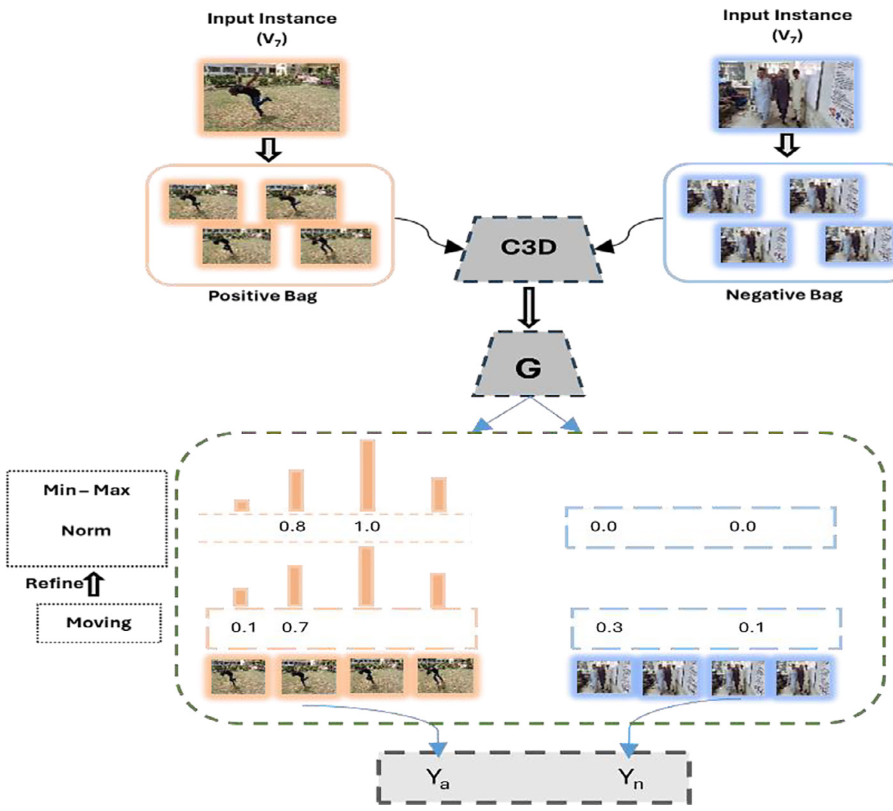

Figure 6. Pseudo labeling generator

minimizing the loss function $\mathcal{L}$ as described in Equation 21:

$$\theta^*_{ESGA} = \arg min_{\theta_{ESGA}} \mathcal{L}\left(ESGA, \hat{Y}^a, \hat{Y}^n\right) \quad (21)$$

where: $\mathcal{L}$ could be a cross-entropy loss or another suitable loss function tailored for self-training and anomaly detection.

To further improve the discriminative ability of the feature encoder ESG_A for anomalous frame detection, a self-guided attention module is incorporated to attend to salient regions of frames. As illustrated in Figure 8, the C3D encoder, enhanced by a self-guided attention mechanism, refines the extracted features before passing them to the classification layer.

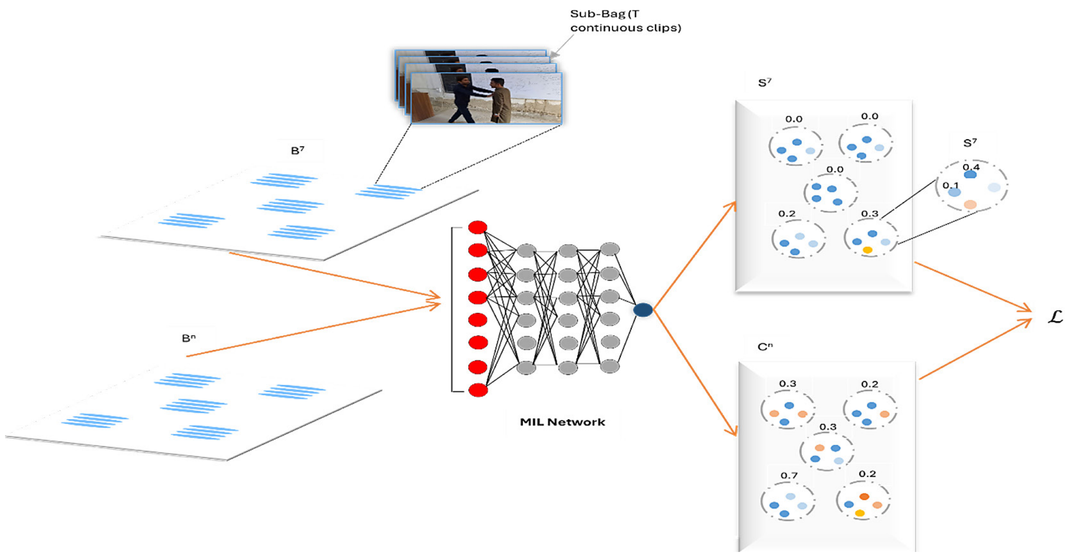

Figure 7. Feature encoder architecture

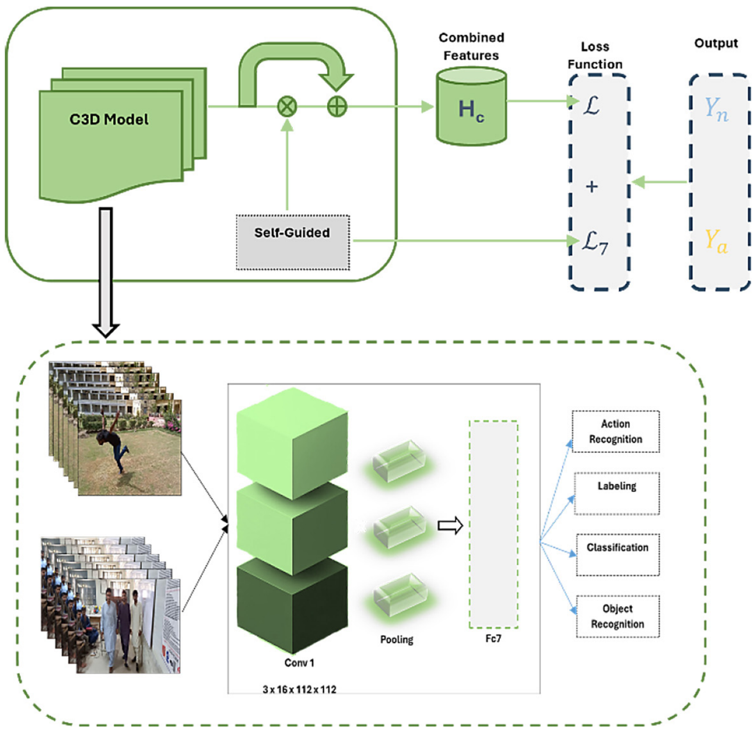

Figure 8. C3D encoder based self-guided attention mechanism

This ensures a smooth transition to the final classification step, where the model determines whether the observed activities are suspicious or non-suspicious.

Classification layer

The classification process begins with the combination of several significant inputs extracted from the Feature Extraction and Labeling process. First, the refined feature vector *frefined*, which contains significant information from the input frames, is obtained from the GAP layer and feature extraction process. This vector is represented in Equation 22:

$$f_{refined} = GAP(F) \tag{22}$$

where: F denotes the feature maps extracted from the video frames.

Henceforth, anomaly scores $\{si\}$ or each frame are generated throughout the pseudo labeling process and play a significant role in generating pseudo labels. The pseudo labels, $\hat{Y} = \{\hat{Y}_i\}$ offer an initial classification of the frames as either anomalous or normal. Subsequently, the self-guided attention mechanism improves the feature representations to become more discriminative and optimal for the classification process.

The refined feature vector *frefined* and the generated pseudo labels $\hat{Y}$ are then passed into the classification layer. The classification process involves a fully connected layer, which performs a linear transformation of the features. This operation is mathematically represented in Equation 23:

$$z = W.f_{refined} + b \tag{23}$$

where: z represents the logits, W is the weight matrix, and b is the bias term. The logits z contain raw prediction values for each class – suspicious and non-suspicious activities.

Next, these logits are fed into the softmax activation function, which normalizes them into a probability distribution across the classes as define in Equation 24:

$$P(y = j \mid z) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{24}$$

where: $P(y = j \mid z)$ denotes the probability of the input being classified as class j, with K representing the total number of classes. This probabilistic output enables the model to assign a confidence score to each classification, allowing for the detection of suspicious activities with a degree of certainty.

To fine-tune the model, cross-entropy loss is calculated as shown in Equation 25:

$$L = -\sum_{j=1}^{K} y_j \log\big(P(y = j \mid z)\big) \tag{25}$$

where: y_j is the ground truth label. This loss function penalizes incorrect classifications and guides the optimization process during training. The backpropagation algorithm then updates the weights W and bias b based on the computed gradients, minimizing the loss and improving the model's classification accuracy over time.

RESULTS

The results section evaluates the performance of the developed algorithm using Dataset-I and Dataset-II. During the training phase, a collective strategy was employed, involving a comprehensive training set of 786,367 images distributed across ten classes. Training utilized a Google Colab T4 GPU with the Adam optimizer (learning rate = 1e−4, batch size = 32, 50 epochs) This includes 16,129 images from Dataset-I and 770,238 images from Dataset-II, representing 70% of the total image count for each dataset, as detailed in (Table2). For testing, Dataset-I comprised 6.912 images, and Dataset-II included 330,102 images, making up 30% of the total image count for each dataset, as shown in (Table 3). These distributions were crucial for ensuring a balanced and comprehensive assessment of the model's accuracy and robustness across varied scenarios. To ensure statistical reliability and minimize variance due to data splits, a 5-fold cross-validation was additionally conducted. The results reported align consistently across folds, validating the model's generalizability.

Table 2. Train data distribution of Dataset-I and Dataset-II

Metric	Kicking	Punching	Running	Normal	Pushing	Smoking	Throwing	Jumping	Falling	Talking
Dataset-I	1921	1505	1477	1890	1111	1463	1365	1890	1610	1897
Dataset-II	57303	91113	55251	57968	110118	44512	64610	114051	66613	108697

Table 3. Test data distribution of Dataset-I and Dataset-II

Metric	Kicking	Punching	Running	Normal	Pushing	Smoking	Throwing	Jumping	Falling	Talking
Dataset-I	823	645	633	810	476	627	585	810	690	813
Dataset-II	24559	39049	23679	24843	47194	19077	27690	48879	28549	46585

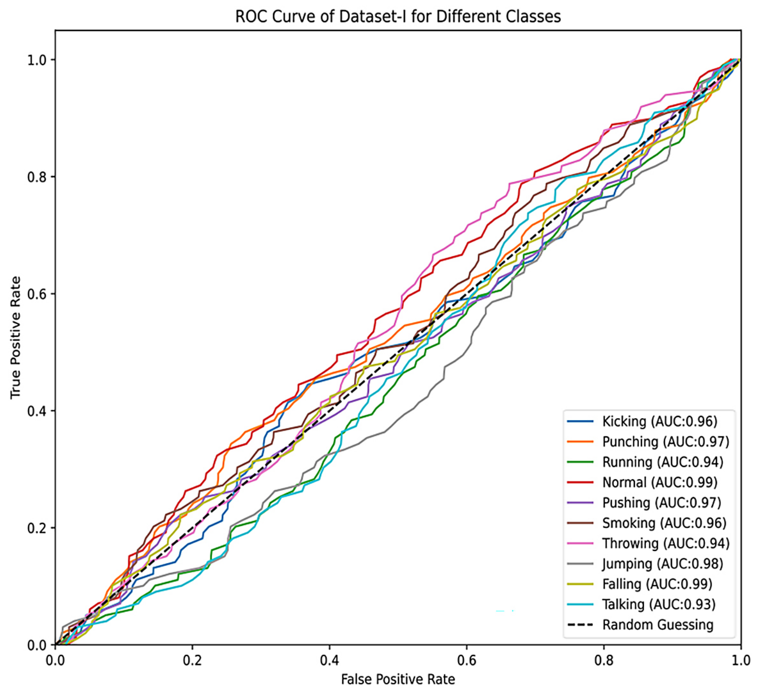

Figure 9. ROC curve analysis of CampusWatch (Dataset-I)

ROC curve analysis across datasets

The ROC curve analysis assesses the model's performance in distinguishing between different classes. Each class is represented by a distinct color on the ROC curve, as depicted in (Figure 9) for dataset-I. AUCs are computed to measure the discriminative ability of the model with respect to each class. The ROC curves for each class are depicted with distinct colors. The AUC values for these classes are summarized in (Table 4). Among the suspicious activity classes, "Jumping", "Falling" and "Normal" demonstrated the strongest performance, each with an AUC value of 0.98 or higher in Dataset-I and 0.94 or higher in Dataset-II. The "Talking" class recorded the lowest AUC values, indicating relatively lower discriminative power for this activity, with AUCs of 0.93 in Dataset-I and 0.89 in Dataset-II.

Confusion matrix analysis across datasets

The confusion matrix provides a detailed breakdown of the model's classification performance across the 10 activity classes. (Figure 10) present the confusion matrices for Dataset-I and Dataset-II, offering insights into where the model performs well and where it faces challenges.

Punching: Across both datasets, the Punching class exhibits highest accuracy, with the majority of predictions being correctly identified. However, there is considerable misunderstanding between the Kicking and Running classes, indicating a moderate overlap in how the model perceives these activities. Throwing: Throwing is significantly misclassified with Kicking and Running in both the datasets. This implies that the model struggles to distinguish between these physically comparable activities, indicating a need for additional refining. Talking: The Talking class consistently demonstrates significant misclassifications across multiple other classes, leading to its lower AUC values, particularly in Dataset II. This highlights the model's difficulty in effectively recognizing Talking and indicates areas for improvement.

Overall, the confusion matrix analysis demonstrates the model's strengths in the "Punching"

Table 4. ROC curve analysis results for Dataset-I and Dataset-II

Metric	Kicking	Punching	Running	Normal	Pushing	Smoking	Throwing	Jumping	Falling	Talking
Dataset-I	0.96	0.97	0.94	0.99	0.97	0.96	0.94	0.98	0.99	0.93
Dataset-II	0.91	0.94	0.90	0.96	0.93	0.94	0.92	0.95	0.94	0.89

and the “Normal” classes; however, some areas, like “Throwing” and “Talking”, still require improvement to enhance classification performance across both datasets.

Performance metrics across datasets

To validate the reliability of HYDPL, 5-fold cross-validation was performed on the training set, achieving an average accuracy of 97.86% across folds (Table 5), confirming consistent performance. Furthermore, as shown in (Table 6), Dataset-I achieved 98% accuracy, 0.97 precision, 1 recall, and 0.98 F1-score, with an FPR of 0.28 and no false negatives (FNR = 0). Dataset-II reported 97% accuracy, 0.97 precision, 0.99 recall, and 0.98 F1-score, with higher FPR (0.39) and FNR (0.005). These results show strong detection performance, with improvement needed in minimizing false positives and negatives for Dataset-II.

DISCUSSION

The results obtained from ROC and confusion matrix analyses offer significant insights into the efficiency and accuracy of the HYDPL algorithm in recognizing suspicious activity when applied to both Dataset-I and Dataset-II.

Performance across datasets

The HYDPL algorithm provides promising results with overall accuracy rates of 98% for Dataset-I and 97% for Dataset-II, showcasing its efficiency in recognizing various activities, including suspicious ones, across diverse environments. The slightly higher accuracy in Dataset-I may be due to a more balanced representation of activity types. This result is comparable to modern practices, especially if traditional machine learning methods or less complex neural networks are used, which often have a low accuracy when working with different datasets [34, 35]. Additionally, the consistency of results across cross-validation folds further supports the robustness of the proposed HYDPL algorithm. Moreover, strengths and limitations of the HYDPL algorithm are demonstrated through class class-wise study. The values of AUC ≥ 0.93 were found for “Jumping”, “Smoking”, and “Falling”, demonstrating a strong discriminative capacity. However, the model performs relatively poorly in classifying the “Talking” activity, achieving an AUC of 0.93 on Dataset-I and 0.89 on Dataset-II. This difficulty might be attributed to the less robust visual cues inherent in talking, which are less distinctive than those of more dynamic activities. Earlier studies in similar contexts have also reported challenges in classifying less visually distinctive activities [36, 37].

Misclassification issues

The confusion matrices show the patterns of misclassification, especially in the “Throwing” and “Talking” classes. Misclassification of “Throwing” as “Kicking” and “Running” implies difficulty in distinguishing between similar physical movements. Significant misclassifications in the “Talking” class reflect the classifier’s difficulty in distinguishing subtle movements. Such issues are consistent with other studies that highlight misclassification problems in HAR

Table 5. Cross-validation of Dataset-I and Dataset-II

Fold	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
1	97.8	96.9	99.1	97.9
2	97.5	96.7	99.0	97.8
3	98.1	97.2	99.3	98.2
4	97.9	97.0	99.2	98.0
5	98.0	97.1	99.1	98.1
Avg ± SD	97.86 ± 0.21	97.0 ± 0.18	99.14 ± 0.11	98.0 ± 0.15

Table 6. Performance metrics

Metric	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	FPR (%)	FNR (%)
Dataset-I	98	97	100	98	0.28	0
Dataset-II	97	97	99	98	0.39	0.5

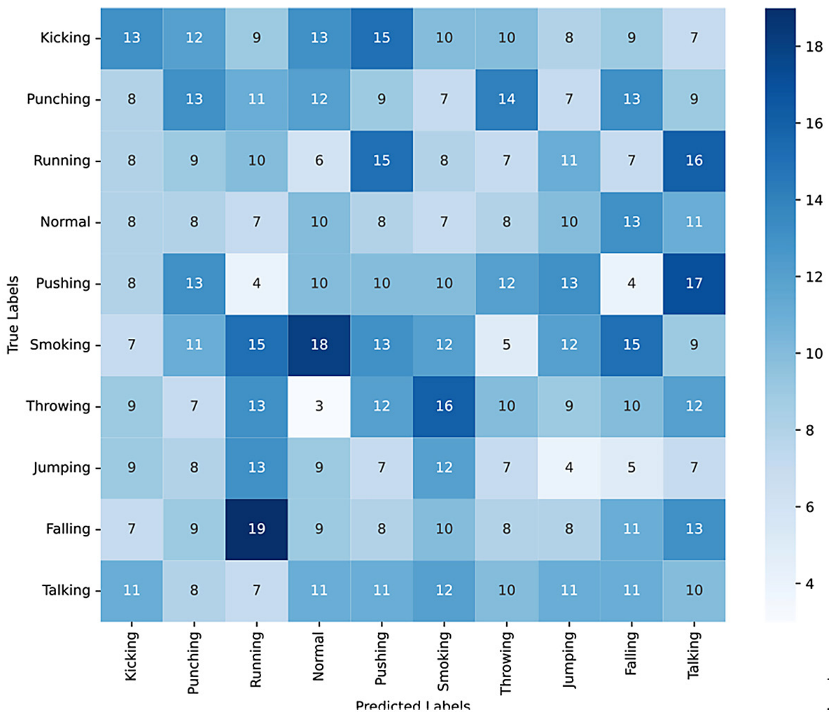

Figure 10. Confusion matrix analysis for each class for Dataset-I

systems, where activity characteristics are often similar, emphasizing the need for model refinement to enhance accuracy [38, 39].

Comparison with other methods

In comparison to other methods, the HYDPL algorithm's overall performance is commendable, especially in handling complex activities and maintaining high accuracy. For instance, traditional machine learning approaches, such as SVM or random forests, typically show lower accuracy when applied to similarly complex tasks [40, 41]. Furthermore, the two-stage self-training scheme implemented in HYDPL significantly enhances the quality of pseudo-labels and anomaly detection, marking an improvement over single-stage training methods commonly used in earlier studies [42–45]. Additionally, (Table 7) compares accuracy rates for the HMDB51 dataset, demonstrating the competitive performance of the HYDPL algorithm against various contemporary methods and highlighting its superior handling of complex activity recognition tasks.

Limitations and future work

The HYDPL algorithm performs well; however, it has certain limitations. For instance, the misclassification of the "Talking" activity shows the model's struggle during classification, indicating the model's need to be further refined with additional features or sophisticated techniques such as attention mechanism or ensemble learning [46–48]. Furthermore, the scope of the study is limited to academia, and only nine suspicious activities are considered. Future research should also explore the impact of incorporating more varied datasets during training to improve the model's adaptability to different environments.

Table 7. Comparison of accuracy rates (%) for HMDB51

Method	Results	Improvement
DNN and VGG19 [49]	0.93	+0.04
CNN with CAMs and AEs [50]	0.77	+0.20
OmniCLIP with PTA and SPG [51]	0.74	+0.23
Dynamic time warping and hierarchical model with K-Class [52]	0.61	+0.36
CNN, ConvLSTM, and LRCN [53]	0.92	+0.05
Two-Stream ConvNet and BiLSTM [54]	0.90	+0.07

In conclusion, the HYDPL algorithm demonstrates strong potential for detecting suspicious activities in academic environments, outperforming several existing approaches in terms of accuracy and robustness. Nonetheless, certain limitations, such as misclassification in cases like "Talking" and the narrow scope of suspicious activity categories, highlight areas for improvement. Future work will aim to integrate advanced techniques, such as attention mechanisms or ensemble models and extend the dataset to enhance generalizability across both academic and non-academic settings.

REFERENCES

1. Kumar P, Chauhan S, Awasthi LK. Human activity recognition (HAR) using deep learning: review, methodologies, progress and future research directions. Arch Comput Methods Eng 2024;31:179–219. https://doi.org/10.1007/S11831-023-09986-X/METRICS
2. Qin Z, Liu Y, Ji P, Kim D, Wang L, McKay RI, et al. Fusing higher-order features in graph neural networks for skeleton-based action recognition. IEEE Trans Neural Networks Learn Syst 2024;35:4783–97. https://doi.org/10.1109/TNNLS.2022.3201518
3. Zhu N, Diethe T, Camplani M, Tao L, Burrows A, Twomey N, et al. Bridging e-health and the internet of things: The SPHERE Project. IEEE Intell Syst 2015;30:39–46. https://doi.org/10.1109/MIS.2015.57
4. Cheoi KJ. Temporal saliency-based suspicious behavior pattern detection. Appl Sci 2020;10:1020. https://doi.org/10.3390/APP10031020
5. Rajpurkar OM, Kamble SS, Nandagiri JP, Nimkar A V. Alert Generation on Detection of Suspicious Activity Using Transfer Learning. 2020 11th Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2020, 2020. https://doi.org/10.1109/ICCCNT49239.2020.9225263
6. Cabanillas-Carbonell M, Rivera JS, Muñoz JS. Artificial intelligence in video surveillance systems for suspicious activity detection and incident response: A systematic literature review. Adv Sci Technol Res J 2025;19:389–405. https://doi.org/10.12913/22998624/196795
7. Concha DT, Maia HDA, Pedrini H, Tacon H, Brito ADS, Chaves HDL, et al. Multi-stream Convolutional Neural Networks for Action Recognition in Video Sequences Based on Adaptive Visual Rhythms. Proc - 17th IEEE Int Conf Mach Learn Appl ICMLA 2018 2018:473–80. https://doi.org/10.1109/ICMLA.2018.00077
8. Chen L, Hoey J, Nugent CD, Cook DJ, Yu Z. Sensor-based activity recognition. IEEE Trans Syst Man Cybern Part C Appl Rev 2012;42:790–808. https://doi.org/10.1109/TSMCC.2012.2198883
9. Lloret J, Canovas A, Sendra S, Parra L. A smart communication architecture for ambient assisted living. IEEE Commun Mag 2015;53:26–33. https://doi.org/10.1109/MCOM.2015.7010512
10. Marsden M, McGuinness K, Little S, O'Connor NE. ResnetCrowd: A residual deep learning architecture for crowd counting, violent behaviour detection and crowd density level classification. 2017 14th IEEE Int. Conf. Adv. Video Signal Based Surveillance, AVSS 2017, Institute of Electrical and Electronics Engineers Inc.; 2017. https://doi.org/10.1109/AVSS.2017.8078482
11. Gorave A, Misra S, Padir O, Patil A, Ladole K. Suspicious activity detection using live video analysis, 2020;203–14. https://doi.org/10.1007/978-981-15-0790-8_21
12. Dhulekar PA, Gandhe ST, Sawale N, Shinde V, Khute S. Surveillance System for Detection of Suspicious Human Activities at War Field. 2018 Int. Conf. Adv. Commun. Comput. Technol. ICACCT 2018, 2018;357–60. https://doi.org/10.1109/ICACCT.2018.8529632
13. Pappula S, Nadendla T, Lomadugu NB, Revanth Nalla S. Detection and Classification of Pneumonia Using Deep Learning by the Dense Net-121 Model. 2023 9th Int Conf Adv Comput Commun Syst ICACCS 2023 2023:1671–5. https://doi.org/10.1109/ICACCS57279.2023.10113110
14. Khan RU, Haq AU, Hussain SM, Ullah S, Almakdi S, Kumar R, et al. Analyzing and Battling the Emerging Variants of Covid-19 Using Artificial Neural Network and Blockchain. 2021 18th Int Comput Conf Wavelet Act Media Technol Inf Process ICCWAMTIP 2021 2021:101–5. https://doi.org/10.1109/ICCWAMTIP53232.2021.9674142
15. Al-qaness MAA, Dahou A, Abd Elaziz M, Helmi AM. Human activity recognition and fall detection using convolutional neural network and transformer-based architecture. Biomed Signal Process Control 2024;95:106412. https://doi.org/10.1016/J.BSPC.2024.106412
16. Kaya Y, Topuz EK. Human activity recognition from multiple sensors data using deep CNNs. Multimed Tools Appl 2024;83:10815–38. https://doi.org/10.1007/S11042-023-15830-Y/METRICS
17. Islam MM, Nooruddin S, Karray F, Muhammad G. Multi-level feature fusion for multimodal human activity recognition in Internet of Healthcare Things. Inf Fusion 2023;94:17–31. https://doi.org/10.1016/J.INFFUS.2023.01.015
18. Kshirsagar AP, Azath H. YOLOv3-based human detection and heuristically modified-LSTM for abnormal human activities detection in ATM machine.

J Vis Commun Image Represent 2023;95:103901. https://doi.org/10.1016/J.JVCIR.2023.103901

19. Sudharson D, Srinithi J, Akshara S, Abhirami K, Sriharshitha P, Priyanka K. Proactive headcount and suspicious activity detection using YOLOv8. Procedia Comput Sci 2023;230:61–9. https://doi.org/10.1016/J.PROCS.2023.12.061
20. Zhou Y. A YOLO-NL object detector for real-time detection. Expert Syst Appl 2024;238:122256. https://doi.org/10.1016/J.ESWA.2023.122256
21. Zhang Y, Ye M, Zhu G, Liu Y, Guo P, Yan J. FFCA-YOLO for small object detection in remote sensing images. IEEE Trans Geosci Remote Sens 2024;62:1–15. https://doi.org/10.1109/TGRS.2024.3363057
22. [22] Kuehne H, Jhuang H, Stiefelhagen R, Serre T. HMDB51: A large video database for human motion recognition. High Perform Comput Sci Eng 2012 - Trans High Perform Comput Center, Stuttgart, HLRS 2012 2013:571–82. https://doi.org/10.1007/978-3-642-33374-3_41
23. HMDB_Human_Activity_Recognition n.d. https://www.kaggle.com/datasets/avigoen/hmdb-human-activity-recognition (accessed November 14, 2024).
24. Gianluigi C, Raimondo S. An innovative algorithm for key frame extraction in video summarization. J Real-Time Image Process 2006;1:69–88. https://doi.org/10.1007/s11554-006-0001-1
25. García GB. Learning Image Processing with OpenCV : exploit the amazing features of OpenCV to create powerful image processing applications through easy-to-follow examples. Packt Publishing; 2015.
26. Jiang Z, Zhao L, Li S, Jia Y, Liquan Z. Real-time object detection method for embedded devices. n.d.
27. Choi W, Heo S. Deep learning approaches to automated video classification of upper limb tension test. Healthc 2021;9. https://doi.org/10.3390/healthcare9111579
28. Malla PP, Sahu S, Alutaibi AI. Classification of Tumor in Brain MR Images Using Deep Convolutional Neural Network and Global Average Pooling. Processes 2023;11. https://doi.org/10.3390/pr11030679
29. Shdefat AY, Mostafa N, Al-Arnaout Z, Kotb Y, Alabed S. Optimizing HAR Systems: Comparative Analysis of Enhanced SVM and k-NN Classifiers. Int J Comput Intell Syst 2024;17:1–32. https://doi.org/10.1007/S44196-024-00554-0/FIGURES/9
30. Culjak, Ivan and Abram, David and Pribanic, Tomislav and Dzapo, Hrvoje and Cifrek M. A brief introduction to OpenCV | IEEE Conference Publication | IEEE Xplore 2012:1725–30. https://ieeexplore.ieee.org/abstract/document/6240859 (accessed August 25, 2024).
31. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ. Densely connected convolutional networks. Proc - 30th IEEE Conf Comput Vis Pattern Recognition, CVPR 2017;2017-January:2261–9. https://doi.org/10.1109/CVPR.2017.243
32. Elnazer M, Elmamoon A, Mustapha AA. A comparative study of deep learning models for human activity recognition. Cloud Comput Data Sci 2025;6:79–93. https://doi.org/10.37256/CCDS.6120256264
33. Rachmatullah MN, Sutarno S, Isnanto RF. Video annomaly classification using convolutional neural network. Comput Eng Appl J 2024;13:74–82. https://doi.org/10.18495/COMENGAPP.V13I1.468
34. Nayak S, Panigrahi CR, Pati B, Nanda S, Hsieh MY. Comparative analysis of HAR datasets using classification algorithms. Comput Sci Inf Syst 2022;19:47–63. https://doi.org/10.2298/CSIS201221043N
35. Dewi C, Chen RC. Human activity recognition based on evolution of features selection and random forest. Conf Proc - IEEE Int Conf Syst Man Cybern 2019;2019-October:2496–501. https://doi.org/10.1109/SMC.2019.8913868
36. Bouchabou D, Nguyen SM, Lohr C, Leduc B, Kanellos I. A survey of human activity recognition in smart homes based on IoT sensors algorithms: taxonomies, challenges, and opportunities with deep learning. Sensors 2021;21:6037. https://doi.org/10.3390/S21186037
37. Kumar P. Human activity recognition with deep learning: overview, Challenges & Possibilities 2021. https://doi.org/10.20944/PREPRINTS202102.0349.V1
38. Fasciglione A, Leotta M, Verri A. Improving activity recognition while reducing misclassification of unknown activities. Proc Work Enabling Technol Infrastruct Collab Enterp WETICE 2021;2021-October:153–8. https://doi.org/10.1109/WETICE53228.2021.00039
39. Lateef R, Abbas A. Tuning the hyperparameters of the 1D CNN Model to improve the performance of human activity recognition. Eng Technol J 2022;40:547–54. https://doi.org/10.30684/ETJ.V40I4.2054
40. Ahmad I, Basheri M, Iqbal MJ, Rahim A. Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection. IEEE Access 2018;6:33789–95. https://doi.org/10.1109/ACCESS.2018.2841987
41. Janiesch C, Zschech P, Heinrich K. Machine learning and deep learning. Electron Mark 2021;31:685–95. https://doi.org/10.1007/S12525-021-00475-2/TABLES/2
42. Zhang C, Li G, Qi Y, Wang S, Qing L, Huang Q, et al. Exploiting completeness and uncertainty of pseudo labels for weakly supervised video anomaly detection 2023:16271–80.
43. Qi Z, Zhu R, Fu Z, Chai W, Kindratenko V. Weakly supervised two-stage training scheme for deep video fight detection model. Proc - Int Conf Tools with Artif Intell ICTAI 2022;2022-October:677–85. https://doi.org/10.1109/ICTAI56018.2022.00105

44. Liu Y, Zhuang C, Lu F. Unsupervised Two-Stage Anomaly Detection 2021.
45. Saleem MH, Velayudhan KK, Potgieter J, Arif KM. Weed identification by single-stage and two-stage neural networks: a study on the impact of image resizers and weights optimization algorithms. Front Plant Sci 2022;13:850666. https://doi.org/10.3389/FPLS.2022.850666/BIBTEX
46. Dudukcu HV, Taskiran M, Gulru Cam Taskiran Z, Kahraman N. Human Activity Recognition with Ensemble Learning. 2024 26th Int Conf Digit Signal Process Its Appl DSPA 2024 2024. https://doi.org/10.1109/DSPA60853.2024.10510025
47. Ghalan M, Aggarwal RK. Novel Human Activity Recognition by graph engineered ensemble deep learning model. IFAC J Syst Control 2024;27:100253. https://doi.org/10.1016/J.IFACSC.2024.100253
48. Haresamudram H, Essa I, Plötz T. Assessing the state of self-supervised human activity recognition using wearables. Proc ACM Interactive, Mobile, Wearable Ubiquitous Technol 2022;6. https://doi.org/10.1145/3550299
49. Khan MA, Javed K, Khan SA, Saba T, Habib U, Khan JA, et al. Human action recognition using fusion of multiview and deep features: an application to video surveillance. Multimed Tools Appl 2024;83:14885–911. https://doi.org/10.1007/S11042-020-08806-9/TABLES/7
50. Dastbaravardeh E, Askarpour S, Saberi Anari M, Rezaee K. Channel attention-based approach with autoencoder network for human action recognition in low-resolution frames. Int J Intell Syst 2024;2024:1052344. https://doi.org/10.1155/2024/1052344
51. Liu M, Li B, Yu Y. OmniCLIP: Adapting CLIP for Video Recognition with Spatial-Temporal Omni-Scale Feature Learning 2024.
52. Wu G, Wen C, Jiang H. Wushu Movement Recognition System Based on DTW Attitude Matching Algorithm. Entertain Comput 2025;52:100877. https://doi.org/10.1016/J.ENTCOM.2024.100877
53. Uddin MA, Talukder MA, Uzzaman MS, Debnath C, Chanda M, Paul S, et al. Deep learning-based human activity recognition using CNN, ConvLSTM, and LRCN. Int J Cogn Comput Eng 2024;5:259–68. https://doi.org/10.1016/J.IJCCE.2024.06.004
54. Butt UM, Ullah HA, Letchmunan S, Tariq I, Hassan FH, Koh TW. Leveraging transfer learning for spatio-temporal human activity recognition from video sequences. Comput Mater Contin 2022;74:5017–33. https://doi.org/10.32604/CMC.2023.035512