Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 11, 1108-1120
2025
Publisher: Learning Gate
DOI: 10.55214/2576-8484.v9i11.11058
© 2025 by the authors; licensee Learning Gate

Human suspicious behavior detection system using deep learning with a closed-circuit television camera

Apisit Seelsat[1*], Prasong Praneetpolgrang[2], Payap Sirinam[3]
[1,3]Office of Graduate Studies, Navaminda Kasatriyadhiraj Royal Air Force Academy, Thailand; apisit.digital@gmail.com (A.S.) p.sirinam@gmail.com (P.S.).
[2]School of Information Technology, Sripatum University, Thailand; prasong.pr@spu.ac.th (P.P.).

Abstract: Automated interpretation of human activities in public surveillance is a pressing need. This study introduces a lightweight behavior detection system combining OpenPose (skeletal keypoints) with a Temporal Convolutional Network (TCN) for sequence analysis. The pipeline is implemented on an NVIDIA Jetson Orin Nano edge device, enabling real-time, privacy-preserving processing of anonymized skeletal data. Experiments on the MPII Human Pose dataset show the TCN model surpasses an LSTM baseline, achieving 91.83% overall accuracy and an 8% improvement on high-velocity actions (e.g., running, punching). A six-class confusion matrix showed a good balance, though walking was the most ambiguous category. Field tests in public spaces, using the model as an edge microservice, validated its real-time performance. The system delivers prompt alerts for safety-critical incidents, such as falls or violence. It demonstrates how edge AI can enhance public safety by reducing delays, minimizing data consumption, and diminishing false alarms. The findings indicate TCNs are proficient in comprehending long-term temporal patterns for behavior detection.

Keywords: *Behavior detection, Deep learning, Temporal data analysis.*

1. Introduction

Nowadays, public surveillance in shopping districts, transit hubs, and airports has become increasingly vital due to rising rates of crime. According to a statistical report from the Royal Thai Police [1], violent incidents in 2023 (B.E. 2566) increased by more than 455% compared with 2022 (B.E. 2565), while murder cases rose by approximately 18.7%. These figures clearly demonstrate a growing safety threat in public areas. Furthermore, the United Nations Office on Drugs and Crime [2] highlights additional Southeast Asian security concerns, such as cybercrime and terrorism, that are expected to exert significant social impacts. These circumstances suggest that human-operated surveillance alone may not be sufficient, especially in densely populated areas or during emergency responses, where timely assessment and response are crucial. Besides crime-based threats, fall incidents among older adults constitute another pressing concern. The World Health Organization [3] identifies falls as a leading cause of serious injury and mortality in the elderly, underscoring the need for timely detection and intervention. Recent advances in artificial intelligence (AI), particularly deep learning (DL), have demonstrated great potential in analyzing complex image and video information. DL methods are more effective than classical techniques in classifying human behavior, specifically when applied to continuous, temporal data streams [4]. However, utilizing these technologies in the real world remains challenging due to factors such as varying lighting, shifting camera angles, and the capability of real-time data processing, which requires high-performance computing systems.

Therefore, this study proposes the development of a model that observes and processes human behavior in public spaces through an integration of OpenPose and Temporal Convolutional Networks (TCN). The model was designed to detect individuals exhibiting abnormal or potentially risky actions.

History: Received: 12 September 2025; Revised: 17 October 2025; Accepted: 22 October 2025; Published: 17 November 2025
* Correspondence: apisit.digital@gmail.com

Its performance was benchmarked against a Long Short-Term Memory (LSTM) approach on the KTH dataset and the MPII Human Pose dataset using standard performance metrics, including accuracy, precision, recall, and F1-score. An extra effort was made to validate the model in real situations. The model was further validated under real-world conditions, and the findings provide practical design guidelines for surveillance systems that enhance reliability and reduce false-alarm rates.

2. Literature Review

2.1. NVIDIA Jetson Orin Nano 8 GB Board

The NVIDIA Jetson Orin Nano 8 GB is a computer module designed for artificial intelligence (AI) at the edge. It features a 6-core Arm Cortex-A78AE CPU and Ampere GPU with 1024 CUDA cores and 32 Tensor Cores, delivering up to 20 TOPS (INT8), expandable to 67 TOPS (Sparse INT8) in "Super Mode" with JetPack SDK 6.2 [5]. With 8 GB LPDDR5 memory and bandwidth up to 102 GB/s, and power configurable between 7–25W, it provides a balance between performance and energy efficiency. It supports CUDA, cuDNN, TensorRT, DeepStream SDK, and is compatible with models built on TensorFlow and PyTorch. This architecture enables on-board inference, reducing communication latency and enhancing privacy by processing data locally.

Its compact size, low power consumption, and high AI performance make it well-suited for applications such as drones, mobile robots, and real-time medical or surveillance systems. The developer kit is affordable (~8,500–9,000 THB), making it an accessible solution for edge-based human behavior detection.

2.2. Human Pose Estimation Algorithm in Computer Vision (OpenPose)

OpenPose, introduced by researchers at Carnegie Mellon University, is a real-time multi-person pose estimation library that uses a bottom-up, two-branch CNN architecture to generate confidence maps and Part Affinity Fields (PAFs) [6]. These enable the system to detect up to 135 body keypoints and accurately assemble skeletons from multiple persons in a single frame.

This framework is widely adopted in applications including sports analytics, rehabilitation, AR/VR tracking, and surveillance. Its efficiency in modeling pose dynamics makes it highly suitable for complex scenes. In surveillance, OpenPose has been integrated into recent research:

- Song et al. [7] utilized OpenPose with a spatial-temporal GCN to detect abnormal pedestrian activity.
- Janbi et al. [8] demonstrated a GCN-based violence detection model using OpenPose, reaching 93% accuracy.
- Nou et al. [9] introduced pose enhancement to improve anomaly tracking and detection using OpenPose skeletons.

Its ability to isolate skeletal structures from background noise and maintain privacy enhances its applicability in safety-critical systems.

2.3. Long Short-Term Memory (LSTM) Neural Networks for Human Behavior Detection

LSTM networks are a subclass of recurrent neural networks (RNNs) capable of modeling long-term temporal dependencies via gated memory cells [10]. This makes LSTMs highly effective for behavior recognition from pose sequences.

LSTM in Skeleton-Based Behavior Detection Skeleton-based methods, using sequences of joint coordinates from pose estimators, reduce background interference and provide temporal structure. LSTMs process this data sequentially to learn motion trajectories. Lin et al. [10] used OpenPose skeletons with LSTM to achieve 98.2% fall detection accuracy in CCTV video. Gatt et al. [11] used LSTM with 1D convolutional autoencoders to detect anomalies in skeletal trajectories.
Other studies introduced enhancements:

- ST-LSTM models joint spatial relationships alongside time dependencies using trust gates to handle occluded/noisy joints.
- GCA-LSTM employed attention to focus on critical joints per action, improving recognition performance.

These innovations emphasize LSTM's adaptability to domain-specific noise, occlusion, and the need for fine-grained discrimination.

2.4. Temporal Convolutional Networks (TCN) in Human Behavior Detection and Surveillance

Temporal Convolutional Networks (TCN) are deep convolutional models employing causal and dilated filters in residual blocks to model long-range sequences efficiently. Compared to RNNs, TCNs benefit from parallelism, stable gradients, and low-latency inference, ideal for real-time surveillance [12].

2.4.1. Applications in Surveillance

- Qi and Sun [12] achieved 99.87% fall detection accuracy using TCN with 3D pose sequences.
- Mittal et al. [13] trained a TCN model on OpenPose skeletons to detect violent behaviors, outperforming LSTM across all classes.
- Yu et al. [14] integrated a TCN with transformer encoders and YOLOv8-Pose, achieving 99.6% accuracy at 19 FPS on edge devices.

2.4.2. Hybrid TCN Architectures

- Hao et al. [15] proposed a Temporal Convolutional Attention Network (TCAN) by incorporating attention mechanisms into TCNs to improve accuracy in sequence modeling.
- Mehta and Yang [16] introduced NAC-TCN, a variant using neighborhood attention for affective video analysis.
- Yan et al. [17] and Shi et al. [18] expanded the architecture by combining TCNs with Graph CNNs to form ST-GCN and 2s-AGCN for spatial-temporal action recognition.

These advancements underscore the power of TCNs in processing complex temporal-spatial patterns, especially when paired with skeletal input, for use in scalable, real-time surveillance systems.

3. Methodology

3.1. System Architecture Design

The proposed system is designed to detect individuals exhibiting atypical behavior through deep learning, following the principles of microservices and AI edge computing. The architecture modularizes the system into independent microservices, promoting flexibility, maintainability, and scalability for future enhancements.

This platform supports real-time detection and alerting of human behavior, employing an edge–server–client paradigm to reduce network latency and maintain GPU-accelerated inference near the camera source. This approach enables efficient processing while preserving responsiveness in deployment environments such as public spaces or surveillance networks. The overall system architecture is illustrated in Figure 1.

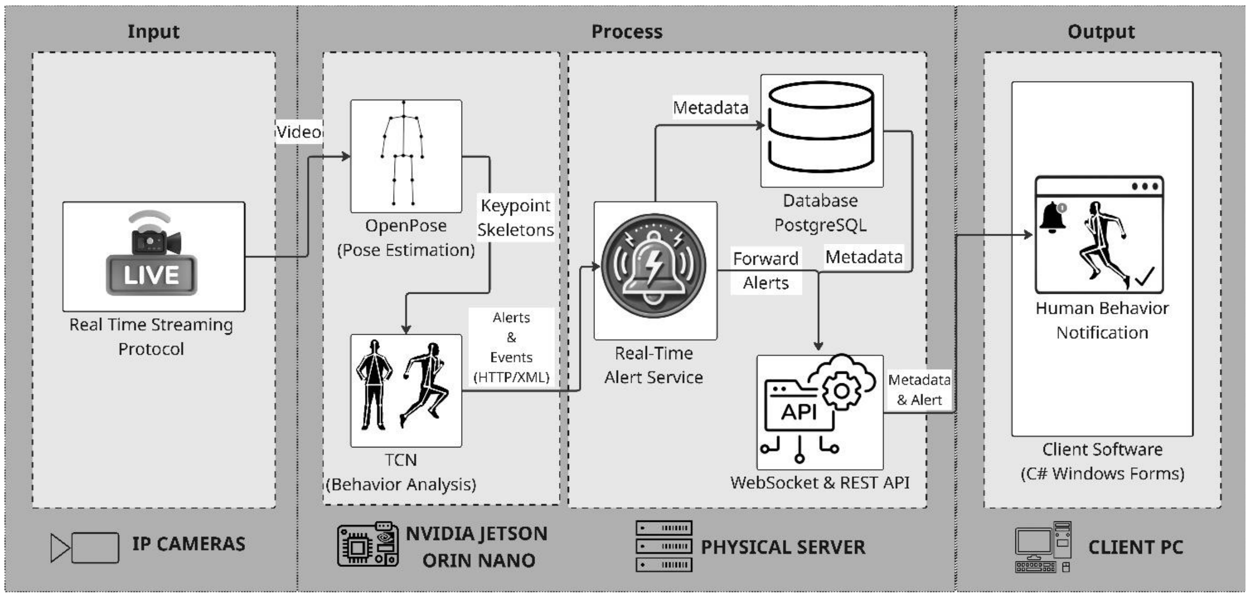

Figure 1.
System Architecture Design.

3.1.1. Hardware Layer

1) IP Cameras. The CCTV acquisition video is captured with a HIKVISION DS-2CD1347G2H-LIU fixed-dome IP camera (4 MP, 25 fps).

2) Edge computing unit. Frames are streamed via an 8-port PoE switch (HIKVISION DS-3E0310HP-E) to an NVIDIA Jetson Orin Nano module (Ampere GPU and Tensor Cores) that is optimized with NVIDIA TensorRT for accelerated inference.

3) Core server. A workstation equipped with an NVIDIA GeForce RTX 4070, an Intel Core i7 CPU, and 32 GB RAM performs model training, data persistence, and API hosting over a 1 GbE LAN backbone.

3.1.2. Software Stack

1) Edge micro-services (Python/TensorFlow and OpenCV).
1.1) Pose Extraction – OpenPose for multi-person 2-D key-point skeletons, all exported to ONNX and compiled into TensorRT engines for real-time throughput.
1.2) Behavior Analysis – Temporal Convolutional Networks (TCN) ingest time-stamped skeleton sequences to classify six behaviors: standing, sitting, falling, walking, running, and punching.
1.3) Notification Service – Publishes HTTP/XML event objects to the server tier. These services interact through internal RESTful endpoints to preserve loose coupling and scalability.
2) Server-side platform (Python and Node.js).
2.1) The Real-Time Alert Service relays incoming alerts to subscribed clients via WebSocket while persisting metadata to PostgreSQL.
2.2) Metadata Logging Server archives images and short video snippets to MinIO-S3 object storage for elastic retention.
3) Client application on Client PC. A Windows Forms dashboard (C#) consumes WebSocket streams to visualize alerts, live thumbnails, and historical queries.

3.1.3. Operational Workflow

The step-by-step operational workflow of the human behavior detection system is summarized in Table 1, outlining the data flow from video capture to behavior classification and notification.

Table 1.
Operational Workflow.

Stage	Data Artefact	Processing Component	Key Operations
Input	RTSP video stream	Jetson Orin Nano	Frame capture & decoding
Pose Estimation	17-keypoint skeletons	OpenPose	Multi-person pose inference
Behavior Classification	Skeleton time-series	TCN	Sliding-window temporal convolutions
Event Packaging	JSON/XML alert	Notification Service	Behavior label, confidence, timestamp
Aggregation	Alert + image hash	Real-Time Alert Service	Persistence (PostgreSQL), broadcast
Output	UI toast + log entry	Windows Forms client	Real-time visual and audible alarm

3.1.4. Data-Flow & Network Considerations

Static IP addressing keeps all edge and camera devices within the same subnet, ensuring deterministic routing. RTSP is tunneled within the LAN to avoid external latency and to satisfy security requirements. The PoE switch unifies power delivery and data transport, mitigating voltage drop and cable clutter while supplying Gigabit uplinks to the server fabric.

3.1.5. Performance Optimization

Jetson-side inference leverages TensorRT INT8 kernels, reducing the model size and boosting throughput from 12 fps to more than 25 fps under laboratory conditions. Batch-size one processing plus on-the-fly affine transformation maintains sub-40 ms frame latency, essential for timely fall detection or anomaly intervention. Dockerized microservices on the server enable horizontal scaling via container orchestration when camera density increases.

3.1.6. Architectural Implications and Summary

By fusing high-resolution IP cameras with an NVIDIA Jetson-powered edge node, a GPU-backed server stack, and a real-time client interface, the system achieves an integrated pipeline for pose-based behavior recognition and instant alert dissemination. The modular microservice design, REST/WebSocket duality, and scalable storage layer collectively provide a robust foundation for deployable surveillance analytics in smart buildings and public safety applications.

The presented architecture illustrates how edge AI can offload computationally intensive pose analytics from the core network, thereby reducing response times for safety-critical scenarios such as elder-fall monitoring or perimeter intrusion. Compared with cloud-centric CCTV analytics, network egress is minimized to lightweight metadata packets, enhancing privacy and lowering bandwidth. Future work will explore federated model updates and introduce self-supervised contrastive learning to address the class imbalance in rare behaviors.

3.2. Data Collection and Dataset Partitioning

3.2.1. Data Collection

This study utilized the MPII Human Pose dataset. The MPII Human Pose dataset contains approximately 25,000 images annotated with joint keypoints for various everyday and athletic activities [19].

3.2.2. Data Partitioning for Training, Validation, and Testing

To prepare the training, validation, and test partitions for human behavior detection with OpenPose, stratified K-Fold cross-validation (K=5) was employed for the MPII Human Pose Dataset.

Step 1 – Stratified K-Fold partitioning. Only video clips depicting the actions of running, walking, sitting, standing, falling, and punching were retained for analysis. These clips were subsequently partitioned using stratified K-Fold cross-validation with K=5, each comprising roughly 200 clips, ensuring that each subset maintains a representative distribution of all action classes.

Step 2 – Cross-validation protocol.

For both datasets, five cross-validation rounds were conducted. In each round, four folds (80%) were used for training the model, while the remaining fold (20%) was used for testing. The test fold was rotated until every subset had functioned exactly once as the evaluation set, ensuring that all data segments contributed to validation at least once.

3.3. Model Training Procedure

Training was conducted in the following steps:

1) Use subsets of data partitioned using K-fold cross-validation.
2) Apply data augmentation (e.g., noise injection, frame shuffling).
3) Input temporal sequences derived from OpenPose into the TCN.
4) Optimize the model using the Cross-Entropy Loss function for multi-class classification.
5) Train initially with the Adam optimizer with learning rate = 0.001. Then, fine-tune with Stochastic Gradient Descent (SGD).
6) Train using mini-batches (e.g., batch size = 32) for each epoch, following these steps:

6.1) Perform forward pass through TCN and LSTM.
6.2) Compute loss.
6.3) Backpropagate gradients.
6.4) Update weights using Adam.
6.5) Monitor validation set accuracy and loss to detect overfitting.

All mini-batches in the training set underwent the same procedure, collectively constituting one epoch. After each epoch, model performance was evaluated on the validation set by computing both accuracy and loss. These metrics provided insight into the model's generalization ability and helped determine whether learning was progressing or signs of overfitting were emerging.

3.4. Feature Extraction

Keypoints produced by OpenPose were transformed into discriminative features through the computation of joint angles, pairwise Euclidean distances, and inter-frame motion vectors.

3.4.1. Joint-Angle Computation

Angular features were derived from keypoint triplets by computing the internal angle formed between two adjacent limb vectors. For instance, the angle at the elbow is calculated using the positions of the shoulder, elbow, and wrist. Given three keypoints A(x1, y1), B(x2, y2), and C(x3, y3), the joint angle can be computed using the cosine law as shown in Equation (1) [20].

$$\text{Angle} = \arccos\left(\frac{(A-B).(C-B)}{|A-B||C-B|}\right) \tag{1}$$

where
$A(x_1, y_1)$: First keypoint, i.e., the shoulder location.
$B(x_2, y_2)$: Second keypoint, i.e., the elbow location.
$C(x_3, y_3)$: Third keypoint, i.e., the wrist location.

3.4.2. Pairwise Euclidean Distance Between Keypoints

Spatial relations were quantified by the straight-line Euclidean distance separating two keypoints in the 2D image plane. The Euclidean distance between two keypoints was calculated using the expression shown in Equation (2) [21].

$$\text{Distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{2}$$

where
$A(x_1, y_1)$: First keypoint (e.g., shoulder position).
$B(x_2, y_2)$: Second keypoint (e.g., elbow position).

3.4.3. Motion-Vector Construction

Temporal dynamics were captured by computing motion vectors, defined as the displacement of each keypoint across successive frames. This method encodes both the orientation and velocity of motion [22]. The motion vector for a given keypoint is represented in Equation (3).

$$\text{Motion Vector} = (x_2 - x_1, y_2 - y_1) \tag{3}$$

Where,
$(x_2 - x_1)$: Difference in the x-coordinates of the same keypoint between the previous and the next frame.
$(y_2 - y_1)$: Difference in the y-coordinates of the same keypoint between the previous and the next frame.

3.4.4. Per-Clip Feature Extraction

For each 500-frame sequence, the following features were computed:

1) Joint angles and their temporal derivatives (angular velocity).
2) Frame-wise Euclidean distances, e.g., ankle-to-knee.
3) Frame-to-frame motion vectors of all tracked keypoints.

3.4.5. Time-Series Construction

Time-Series Data Construction was carried out through the following steps:

1) Keypoint coordinates from successive frames were concatenated to form temporal sequences.
2) The sequences represented motion patterns such as walking, running, or falling.
3) All coordinates were normalized to the range [0,1] to mitigate scale disparities.
4) Behavior labels were assigned for model training, comprising six classes: standing, sitting, falling, walking, running, and punching.

3.5. Implementation of Temporal Convolutional Networks Methodology for Model Training

The steps for TCN deployment comprised:

1) Feature Extraction and Sequence Construction

Sequential features, including joint angles, Euclidean distances between keypoints, and motion vectors, were extracted and compiled into time-series data for TCN processing.

2) Optimization Algorithm and Hyperparameters Configuration

2.1) Optimizer: SGD with Adam was the primary optimization strategy.

2.2) Hyperparameters: Learning rate, batch size, and the number of training epochs were adjusted based on dataset attributes and validation results.

3) Refinement

3.1) Layer Freezing: When using pre-trained components (e.g., OpenPose or feature extraction modules), initial layers were selectively frozen to focus training on TCN blocks and classification layers.

3.2) Early Stopping: Training was terminated if validation loss failed to improve for 5–10 consecutive epochs, thereby mitigating the risk of overfitting.

3.6. Behavior Classification

By integrating OpenPose with TCN, the system generated keypoint coordinates and corresponding confidence scores for six behavior classes: standing, sitting, falling, walking, running, and punching. Each behavior was categorized according to spatial keypoint patterns.

The visualization of keypoints in each pixel generated by OpenPose, mapped onto the human body for each behavior, is shown in Figure 2.

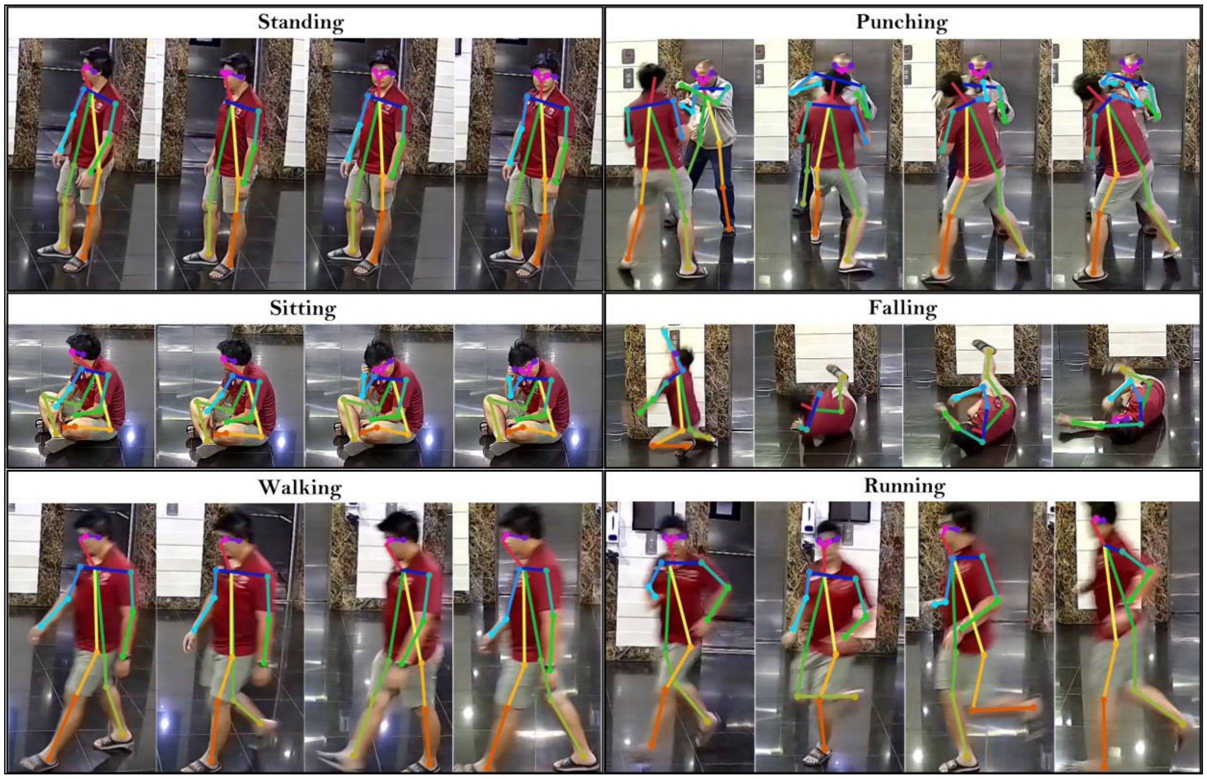

Figure 2.
Keypoints generated by OpenPose in each pixel on the human body of each behavior.

3.7. Prototype deployment in a real-world scenario

1) Model Preparation. The OpenPose model was integrated with TCN in PyTorch, exported to ONNX, and subsequently optimized with the OpenCV DNN module and the TensorRT engine for real-time inference on an NVIDIA Jetson Orin Nano.

2) Installation of hardware and connectivity. Cameras for the Jetson Orin Nano were configured with Ubuntu 18.04, along with CUDA and TensorRT libraries. The IP camera (Hikvision DS-2CD1347G2H-LIU) was connected through an eight-port PoE switch (Hikvision DS-3E0310HP-E). Static IP addresses were assigned to all devices on the network, which were connected to the server and AI edge processors based on the outlined system architecture design topics. This arrangement follows AI edge computing principles, minimizing latency and enhancing cybersecurity by segregating the processing network from the public Internet and enforcing encrypted data transmission.

3) System execution. Video streams were processed frame by frame. OpenPose extracted keypoints, which were then forwarded to the TCN for behavior classification.

4) Human behavior detection & classification. The AI edge processor continuously analyzed video frames streamed from the IP camera to detect the presence of human subjects. Upon detection, the processor extracted relevant keypoint features and inferred the associated behavior. When an observed action matched one of the model's trained categories standing, sitting, falling, walking, running, or punching the event was recognized, and a snapshot was captured immediately. An alert was then issued by the server in accordance with the predefined notification policy, which prioritized behaviors based on their frequency and severity (e.g., punching or running). When such behaviors were detected, the

system automatically dispatched a notification and displayed the corresponding captured image. The process of human behavior detection and classification is illustrated in Figure 3.

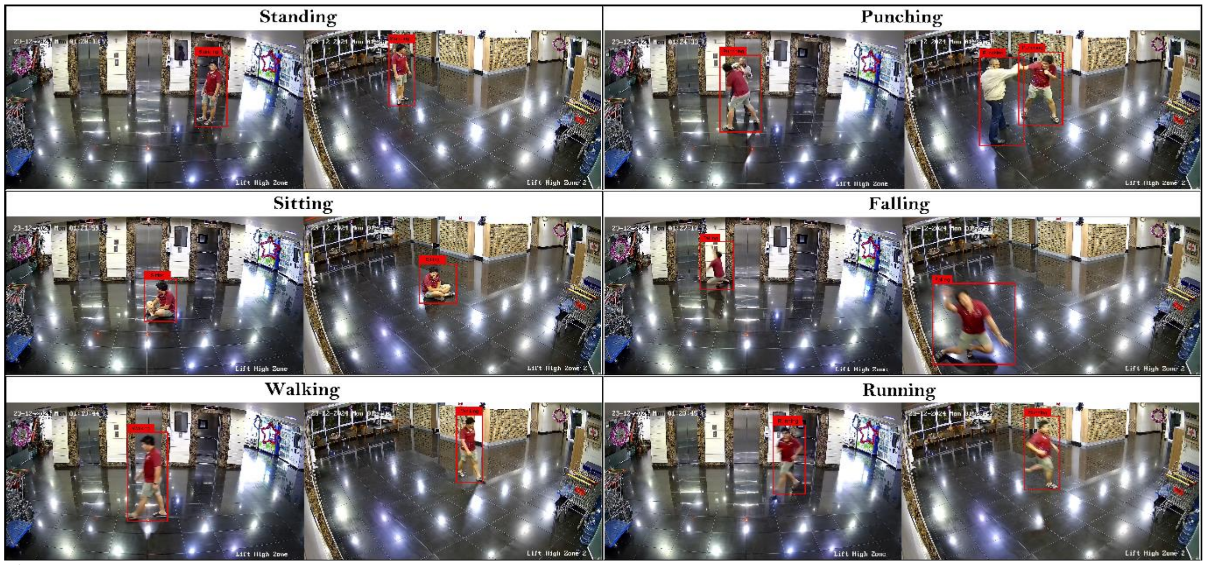

Figure 3.
The human behavior detection & classification.

3.8. Ethical Considerations

This study utilized publicly available and anonymized datasets (e.g., MPII Human Pose dataset) for training and evaluation. Field tests, as described in section 3.7, were conducted on public camera feeds without recording or identifying individuals. Therefore, formal Institutional Review Board (IRB) approval was not required.

4. Results

4.1. Evaluation of Model Performance in Human Behavior Detection

The detailed experimental results and comparison of accuracy, precision, recall, and F1-score for different models are illustrated in Table 2.

Table 2.
Comparison of accuracy, precision, recall, and F1-score for different models.

Behavior	Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Standing	OpenPose + LSTM	92	91	93	92
	OpenPose + TCN	**98**	**94**	**96**	**95**
Sitting	OpenPose + LSTM	90	89	91	90
	OpenPose + TCN	**97**	**93**	**95**	**94**
Falling	OpenPose + LSTM	89	88	90	89
	OpenPose + TCN	**93**	**92**	**94**	**93**
Walking	OpenPose + LSTM	85	84	87	85
	OpenPose + TCN	**91**	**90**	**92**	**91**
Running	OpenPose + LSTM	80	79	82	80
	OpenPose + TCN	**88**	**87**	**89**	**88**
Punching	OpenPose + LSTM	78	77	79	78
	OpenPose + TCN	**85**	**84**	**87**	**85**

4.2. Comparative Analysis of Models

4.2.1. Accuracy

The OpenPose combined with the TCN model surpassed the OpenPose with LSTM across all behavioral categories, yielding an average accuracy enhancement of 3%–8%. The advantages of integrating OpenPose with TCN were most evident in complex, swift movements such as running and punching. Although the LSTM model achieved commendable accuracy for basic behaviors such as standing and sitting, with an average accuracy of around 90%, its performance dropped significantly for rapid-movement classes.

4.2.2. Precision

The OpenPose combined with the TCN model exhibited superior precision scores for all behavior classes when compared to the OpenPose combined with the LSTM model. This was particularly evident in high-velocity actions such as punching, where TCN achieved approximately 7% higher precision. In contrast, the OpenPose-LSTM model demonstrated lower precision in behaviors requiring complex temporal analysis, such as falling and running.

4.2.3. Recall

The OpenPose combined with the TCN model exhibited enhanced recall in intricate behaviors such as walking, running, and punching, indicating superior efficacy in retrieving pertinent behavioral instances. In contrast, the OpenPose–LSTM model showed lower recall for behaviors that rely heavily on tracking swift positional alterations, with particularly diminished performance observed in the running class.

4.2.4. F1-Score

The OpenPose combined with the TCN model produced superior F1-scores compared to the OpenPose combined with the LSTM model across all behavior classes. Notably, improvements of approximately 7% to 8% were observed in high-velocity actions such as running and punching. While the OpenPose–LSTM model maintained strong performance in simpler behaviors like standing and sitting, it exhibited lower F1-scores in more complex, dynamic behaviors.

4.3. Summary of Model Performance for Human Behavior Detection

The OpenPose combined with the TCN model demonstrated outstanding effectiveness in recognizing complex and swift-motion behaviors due to its proficiency in analyzing temporal sequences. It achieved a high degree of accuracy and maintained a good balance between precision and recall. Although the OpenPose combined with the LSTM model exhibited lower overall performance, it proved more suitable for simple, stationary behaviors that require minimal temporal analysis, such as standing and sitting. Therefore, model selection should be guided by the specific operational requirements of the surveillance system. For applications requiring rapid and accurate detection of complex human behaviors, the TCN-based approach offers superior reliability and responsiveness, making it especially well-suited for real-time surveillance scenarios.

4.4. Summary of Model Performance Assessed by a Confusion Matrix

The classifier was evaluated across six behavior classes, each comprising 100 ground-truth instances. The cross-tabulation of true labels versus predicted labels is presented in Table 3, illustrating the classification outcomes for each behavior class.

Table 3.
Confusion matrix for the six human behavior classes.

Actual \ Predicted	Standing	Sitting	Falling	Walking	Running	Punching
Standing	98	0	1	0	0	1
Sitting	0	97	3	0	0	0
Falling	5	0	93	2	0	0
Walking	2	0	2	91	3	2
Running	0	0	0	11	88	1
Punching	4	0	0	10	1	85

The information presented in Table 3 can be summarized as follows:

4.4.1. Overall Accuracy

The model attained an overall accuracy of 91.83%. This indicates a robust general performance of the prediction model.

4.4.2. Precision Analysis

Precision exhibited variability among classes. In the sitting class, the highest precision (100%) was achieved, indicating the complete absence of false positives. In contrast, the lowest precision was observed on walking, as it was frequently misclassified as running or punching.

4.4.3. Recall Analysis

Recall was highest for the basic postures of standing and sitting. This suggests that the model was most effective at detecting behaviors involving relatively simple and static physical movements. In contrast, running and punching exhibited the lowest recall, likely due to overlapping movement patterns or insufficient training samples in these more dynamic and complex behavior categories. This disparity highlights the challenges of accurately recognizing high-velocity actions with limited or ambiguous temporal cues.

4.4.4. Error Analysis

- Standing: Two errors were misclassified once as falling and once as punching.
- Sitting: Three misclassifications, all erroneously predicted as falling.
- Falling: Five instances predicted as standing and two as walking.
- Walking: Confused with standing (2), falling (2), running (3), and punching (2).
- Running: Misidentified as walking on eleven occasions and as punching once
- Punching: Mislabeled as standing (4), walking (10), and running (1).

5. Discussion

This study demonstrates that the integration of OpenPose for skeletal feature extraction with a Temporal Convolutional Network (TCN) for sequence modeling yields significant improvements in Human Activity Recognition (HAR), achieving 91.83% overall accuracy across six target behaviors. The findings confirm that this pose-based temporal framework effectively captures complex motion dynamics. The TCN architecture proved particularly adept at processing activities with gradual state transitions (e.g., falling) and distinguishing high-tempo movements (e.g., running), thereby reducing inter-class confusion. The system maintains a favorable speed-accuracy balance, validating its suitability for latency-sensitive, real-time surveillance tasks.

Despite this performance, limitations exist. The model showed some confusion between high-tempo or kinematically similar motions (e.g., "walking" vs. "running", "punching"), indicating a need for denser joint information and more extensive training data for these specific classes. Nonetheless, the framework demonstrates strong application potential for security and safety, such as real-time alerts for anomalous events ("punching", "falling") in public surveillance or automated patient monitoring.

Future work must prioritize model robustness and generalizability. This requires significant diversification of training data, encompassing varied subjects, environments, and more complex activities. Key strategic enhancements should include the adoption of multi-view inputs to mitigate occlusion and multi-modal data (e.g., LiDAR, IMU) to add discriminative depth or motion cues. Architecturally, a hybrid cloud-edge deployment is recommended to ensure a scalable and responsive system, leveraging the cloud for intensive training and edge devices for low-latency, privacy-preserving inference.

6. Conclusions

This research presented a system for recognizing suspicious behaviors in CCTV video, integrating OpenPose with a TCN to capture spatio-temporal motion dynamics. The system successfully classified six key behaviors with an overall accuracy of 91.83% and demonstrated real-time capabilities suitable for latency-sensitive surveillance missions. The results confirm that utilizing pose-based features in conjunction with a temporal sequence model enhances situational awareness and reduces time-to-response in critical scenarios.

Despite existing limitations related to data diversity and the complexity of high-tempo activities, the proposed system establishes a solid technical foundation for real-world intelligent surveillance. Key future directions include expanding dataset diversity, integrating multi-view and multi-modal data, and implementing a hybrid cloud-edge architecture to achieve an operational balance of accuracy, efficiency, and privacy.

Transparency:

The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained. This study followed all ethical practices during writing.

Acknowledgements:

The authors would like to express their sincere gratitude to Prof. Dr. Prasong Praneetpolgrang and Dr. Payap Sirinam for their invaluable guidance, insightful suggestions, and continued support throughout this research. This work was conducted in partial fulfillment of the requirements for the Master's Degree at the Office of Graduate Studies, Navaminda Kasatriyadhiraj Royal Air Force Academy.

Copyright:

References

[1] Royal Thai Police, "Annual report of criminal statistics and crime situation B.E.2566," 2023. http://thaicrimes.org/crimestat. [Accessed 18-Feb-2024]
[2] United Nations Office on Drugs and Crime, *Global study on homicide 2020: Trends and patterns*. Vienna, Austria: UNODC, 2021.
[3] World Health Organization, "Falls: Key facts. WHO Fact Sheet," 2018. https://www.who.int/news-room/fact-sheets/detail/falls. [Accessed 17-Jan-2024]
[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, 1st ed. Cambridge, MA, USA: MIT Press, 2016.
[5] S. H. Sheshadri, L. S. Karumbunathan, and D. Franklin, "NVIDIA Jetson Orin Nano developer Kit gets a 'Super' boost. NVIDIA Technical Blog," 2024. https://developer.nvidia.com/blog/nvidia-jetson-orin-nano-developer-kit-gets-a-super-boost. [Accessed 09-Nov-2025]
[6] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 2021. https://doi.org/10.1109/TPAMI.2019.2929257

[7] G. Song, Y. Qian, and Y. Wang, "Stgcn-pad: a spatial-temporal graph convolutional network for detecting abnormal pedestrian motion patterns at grade crossings," *Pattern Analysis and Applications*, vol. 28, p. 2, 2025. https://doi.org/10.1007/s10044-024-01382-w

[8] N. F. Janbi, M. A. Ghaseb, and A. A. Almazroi, "ESTS-GCN: An ensemble spatial–temporal skeleton-based graph convolutional networks for violence detection," *International Journal of Intelligent Systems*, vol. 2024, no. 1, p. 2323337, 2024. https://doi.org/10.1155/2024/2323337

[9] S. Nou, J.-S. Lee, N. Ohyama, and T. Obi, "Human pose feature enhancement for human anomaly detection and tracking," *International Journal of Information Technology*, vol. 17, pp. 1311-1320, 2025. https://doi.org/10.1007/s41870-024-02363-2

[10] C.-B. Lin, Z. Dong, W.-K. Kuan, and Y.-F. Huang, "A framework for fall detection based on OpenPose skeleton and LSTM/GRU models," *Applied Sciences*, vol. 11, no. 1, p. 329, 2020. https://doi.org/10.3390/app11010329

[11] T. Gatt, D. Seychell, and A. Dingli, "Detecting human abnormal behaviour through a video generated model," in *2019 11th International Symposium on Image and Signal Processing and Analysis (ISPA). IEEE*, 2019.

[12] L. Qi and H. Sun, "Multistage fall detection framework via 3D pose sequences and TCN integration," *Scientific Reports*, vol. 15, p. 27832, 2025. https://doi.org/10.1038/s41598-025-11325-y

[13] S. Mittal, A. Gupta, and A. R. Chowdhury, "Pose-based violence detection in surveillance videos using temporal convolutional networks," *Pattern Recognition Letters*, vol. 164, pp. 1–8, 2023.

[14] X. Yu, C. Wang, W. Wu, and S. Xiong, "A Real-time skeleton-based fall detection algorithm based on temporal convolutional networks and transformer encoder," *Pervasive and Mobile Computing*, vol. 107, p. 102016, 2025. https://doi.org/10.1016/j.pmcj.2025.102016

[15] H. Hao, Y. Liu, and L. Zhang, "Temporal convolutional attention network for sequence modelling," presented at the Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Barcelona, Spain, 2020.

[16] A. Mehta and W. Yang, "NAC-TCN: Neighbourhood attention temporal convolutional networks for affective video understanding," in *Proc. Int. Conf. Video Image Process. (ICVIP), Shanghai, China*, 2023.

[17] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[18] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[19] U. Iqbal and J. Gall, "Multi-person pose estimation with local joint-to-person associations," in *European Conference on Computer Vision (pp. 627-642). Cham: Springer International Publishing*, 2016.

[20] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition," *Journal of Visual Communication and Image Representation*, vol. 25, no. 1, pp. 24-38, 2014. https://doi.org/10.1016/j.jvcir.2013.04.007

[21] X. Yang and Y. Tian, "EigenJoints-based action recognition using Naïve-Bayes-Nearest-Neighbor," presented at the 2012 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (pp. 14–19). Providence, RI, USA, 2012.

[22] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *2008 IEEE Conference on Computer Vision and Pattern Recognition (pp. 1-8). IEEE*, 2008.