Journal of Soft Computing and Computer Applications

Manuscript 1021

Real-Time Hand Gesture Recognition System for Abductees Rescue Using Deep Learning Techniques

Aws Saood Mohamed

Nidaa Flaih Hassan

Abeer Salim Jamil

ORIGINAL STUDY

Real-Time Hand Gesture Recognition System for Abductees Rescue Using Deep Learning Techniques

Aws Saood Mohamed [a,*], **Nidaa Flaih Hassan** [b], **Abeer Salim Jamil** [c]

[a] University of Technology - Iraq, College of Computer Science, Al-Sina'a St., Al-Wehda District, 10066 Baghdad, Iraq
[b] University of Technology - Iraq, College of Computer Science, Department of Computer Security and Cybersecurity, Al-Sina'a St., Al-Wehda District, 10066 Baghdad, Iraq
[c] Al-Mansour University College, Department of Computer Technology Engineering, Nidhal St., Near Al-Andalus Sq., 10066 Baghdad, Iraq

ABSTRACT

Hand gesture recognition is a challenging problem in computer vision, particularly in terms of security surveillance applications. This study presents the first efficient system for abduction-related hand gesture real-time detection based on deep learning. The most critical problem is to detect and recognize hand gestures in real surveillance conditions and to be computationally effective for real-time multi-hand tracking in various lighting situations while allowing reliable surveillance beyond the 1–4 meters limitation. The proposed system consists of three main parts: The adaptive hand tracking algorithm, which has been used to create the Abductees-Rescue dataset. Introduced pose estimation You Only Look Once version eight, the MediaPipe Hands framework, which variable 3D landmarks extraction at distance. Adaptive Long Short-Term Memory, which was produced by an optimized LSTM model. It has been implemented with the help of the multi-threaded notification approach. All three parts are combined to create novel approaches to the field of abduction rescue research. The Abductees-Rescue dataset includes 9,111 samples, 4,545 native and 4,566 abduction-related 3D hand landmarks. The optimized LSTM model multiplies into a hierarchical signal flow with L2-regularization and an additional dropout approach of achieving 96.12% classifying efficiency. Multi-threaded notification has been implemented, allowing for more than two hand-tracking at the same time, all for being 15 FPS sometimes. Pose estimation has been adapted to temporal detection that allows 10-meter-radius action from perpetrators. It utilizes the MediaPipe Hands framework adapted to 3D landmark actions in the 10 meters distance. Integrated with the Telegram Bot API to enable the multi-threaded engineers to send messages with a picture of abduction delivery time, 1.839 seconds delivery time. This system is a significant advance in building surveillance-based gesture recognition for assurance.

Received 17 May 2025; accepted 3 November 2025.
Available online 26 December 2025

* Corresponding author.
E-mail addresses: cs.21.06@grad.uotechnology.edu.iq (A. S. Mohamed), 110020@uotechnology.edu.iq (N. F. Hassan), abeer.salim@muc.edu.iq (A. S. Jamil).

https://doi.org/10.70403/3008-1084.1021

Keywords: Abduction rescue, Hand gesture recognition, YOLOv8 pose estimation, Long short-term memory, MediaPipe hands framework

1. Introduction

Hand gesture recognition is an important research area of computer vision, having applications in the fields of human-computer interaction, sign language translation, and security embedded systems [1]. Similar to other biometric identification systems, such as face recognition, which has shown significant advancement in security surveillance applications [2], gesture recognition systems provide unique opportunities for non-intrusive identification and monitoring. Detection of the distress signal through hand gestures in security applications provides a subtle way for individuals to request help during critical situations. The gestures of "Signal for Help," created by the Canadian Women's Foundation in 2020, is a specific hand signal that allows victims of violence or abduction to silently signal that they need help without alerting potential perpetrators [3].

The implementation of hand gesture recognition systems in surveillance environments involves specific technical challenges different from those in controlled environments. These challenges cover variable lighting conditions, complex backgrounds, occlusions, and the detection under distances standard to the surveillance camera domains [4].

For example, the most recent hand detection frameworks such as MediaPipe Hands; which its output "hand landmarks" are used in the process of hand gesture recognition, provide high accuracy at close distances as shown in the original work, but in practice it suffers from degradation in performance at distances exceeding 2 meters, constricting their applicability in surveillance systems where cameras coverage generally works on an extended distance to 8 meters and more [5].

Recent studies have explored the deep learning-based methods for hand gesture recognition, achieving varying degrees of success in addressing important challenges like spatial variability and temporal dynamics. The Convolution Neural Network (CNN) architecture has been demonstrated as successful for spatial feature extraction for hand images and therefore utilizes shape and pose information [6, 7].

In another research, the effective integration of deep learning techniques for real-time visual detection tasks was demonstrated, achieving stable performance at 15 Frames Per Second (FPS) through optimized processing pipelines. These studies underscore the technical challenges in developing practical gesture recognition systems for surveillance applications. At the same time, Long Short-Term Memory (LSTM) networks have demonstrated significant capabilities for learning temporal dependencies within the gesture sequences, allowing for the recognition of dynamic hand motion across time [9].

There are limited studies that have investigated the recognition of the "Signal for Help" gesture in surveillance settings. In research [10], authors proposed a dual-CNN architecture for Signal for Help detection, which achieved 91.25% accuracy, yet their approach had a limited distance of less than 4 meters in front of the camera with a small range of conditions in their dataset. Similarly, in research [11], the used approach achieved 98.79% accuracy using deep neural networks; they limited their evaluation to close ranges, close-up hand regions with perceived visible and clear hand regions in consistent illumination conditions. Such constraints render existing methods limited to applicable and not generalizable in authentic surveillance environments. Such constraints render existing methods limited to applicable

and not generalizable in authentic surveillance environments. However, their approaches have been tested on near (1 to 2 meters) acquired datasets under controlled illuminations, with mobile devices or laptop cameras instead of surveillance cameras [12].

The present paper aims to solve these limitations by introducing a system for real-time recognition of abduction-related hand gestures during people's movement in an area equipped with surveillance cameras. The proposed system consists of three main components:

- Designing the Rescue-Abductees dataset: This process includes recording videos via security camera mimicking real scenarios with volunteer participants, and developing an extended-range hand tracking approach by using You Only Look Once version eight (YOLOv8) pose estimation model to create adaptive bounding boxes that provide hand detection and tracking in fixed dimensions to extract the 3D hand landmarks then; via applying the MediaPipe Hands framework, for distances of more than 10 meters.
- Building and training an optimized LSTM model: The model would be trained on the Abductees-Rescue dataset to classify and recognize the hand gestures that refer to the Signal for Help.
- Deploying the optimized model with the real-time notification system: The model deployment consists of implementing a multi-threaded processing pipeline that maintains parallel extraction of hand landmarks from the detected hand regions using the MediaPipe Hands framework, which processes the entire hands that appear in the frame at the same time, overcoming the framework's limitations of detecting only two hands as maximum per frame. In addition, applying the LSTM model on the landmarks queue of each detected hand of the sequenced frames in parallel to classify the hand gestures and sending a notification with a photo to the related parties via Telegram bot in real-time when recognizing abduction-related gestures.

The combination of these elements into one system would effectively respond to the three major problems to be solved for hand gesture recognition in surveillance situations: extended-range detection and tracking, environmental dynamicity, and real-time operation with alert notification. This presents a real-world use case for technology that enhances public safety by automatically detecting and notifying potential abduction.

Based on the work reported in the previous sections, develop a robust hand gesture recognition process for security and safety operations. The Following Are the Contributions of This Research:

– **Comprehensive Surveillance-Oriented Dataset:** Create a dataset of hand gestures during abduction called Abductees-Rescue. The hand gesture dataset in this paper is created with 9,111 samples with real-time surveillance camera with monitoring metric distance of 10m+:
– **Hand Tracking Method**: Implement an improved version of YOLOv8-based adaptive bounding based hand tracking which caters for extended reliable hand detection beyond 3-4 meters.
– **Optimized Temporal Gesture Classification**: A novel Hierarchical LSTM model is used to classify the normal hand action and actions during abduction with an accuracy of 96.12%.
– **Real-Time Threaded Processing**: Develop a system which can track 3 hands from the source at a distance up to 14 meters and at 15 FPS based on MediaPipe Hand.
– **Integrated Notification System**: Develop a RESTful asynchronous notification service by Telegram API which can send a notification in just about 1 to 2 seconds.
– **Comprehensive Exit Week Protocol Report**:

This paper outline is divided as follows: an Introduction that will give a brief overview of real-world applications. Section 2: Related work and limitations. Section 3: Methodologies: that includes a) Dataset and Labels. b) Data Pre-processing c) Creation of an LSTM model and training. d) Deploying the optimized models with real-time notification system. Section 4 presents the results and discussion. Finally, Section 5 concludes the paper with a summary of contributions and suggestions for future work.

2. Related works

The field of hand gesture recognition has developed from using classical computer vision methodologies to utilizing deep learning techniques. This transition represents the wider developments in artificial intelligence and computer vision. In addition, it represents an attempt to address some of the ongoing issues in applying it in the real world. This section reviews the relevant research in some domains of hand gesture recognition, emphasizing research on surveillance applications and distress signal detection.

The classical hand gesture recognition systems relied mainly on traditional computer vision methods based on feature extraction and traditional machine learning methods. Authors in research [13] introduced a system for vision-based hand tracking that used an altered Kanade-Lucas-Tomasi (KLT) algorithm with Camshift tracking to achieve recognition rates over 99% on typical gestures. However, their assessment was limited to indoor settings and at close distances. Similarly, in research [14], a Kalman filter-based method for hand trajectory tracking via Microsoft Kinect showed stable tracking under controlled conditions, but its performance rapidly declines outside of the designed environments and at distances of more than 2 meters from the camera.

The capabilities of hand gesture recognition have been revolutionized with deep learning techniques. Authors in research [15] employed CNN architectures for American Sign Language (ASL) recognition, reporting accuracies between 99.90%–100% over four datasets. However, these results were obtained in constrained experimental settings with little variability in the illumination, background complexity, and camera pose, with close distances under 2 meters. Therefore, these parameters are different from those encountered in surveillance applications. In research [16], authors proposed an Enhanced Dense connected CNN (EDenseNet) to classify hand gestures, which obtained an accuracy of 99.64 % across three benchmark datasets due to better feature propagation via improved dense transition layers. While their approach achieved high accuracy, their evaluation was restricted to a close-distance detection from the camera with controlled lighting.

Models that consider temporal information using LSTM-based models have demonstrated good performance in capturing the sequential aspects of hand gestures. In research [17], authors proposed a modified LSTM architecture for sign language recognition based on leap motion controllers and achieved an average accuracy of 72.3% and 89.5% for signed sentences and isolated sign words, respectively. The method proved robust to execution speed variations, although experimentation was limited to the lab, at distances not to exceed 2 meters. In research [9], authors proposed a CNN-Bidirectional LSTM (BiLSTM) architecture with attention mechanisms for the recognition of data glove-based gestures, and the model achieved 95.05% accuracy over seven dynamic gestures. The implementation of their approach was selectively weighted informative temporal segments within gesture sequences; however, the use of wearable sensors makes it hard to apply directly to non-contact monitoring tasks.

Following the introduction of the “Signal for Help” gesture during the COVID-19 pandemic, research specifically addressing the gesture itself has emerged. In research [10]

introduced a dual-CNN framework was introduced for Signal for Help detection, achieving an accuracy of 91.25% on their custom dataset at 16 FPS. However, their framework was tested under a controlled environment at distances beyond 4 meters using a mobile phone camera at eye level, which raises concerns about its applicability in real environments where cameras a mounted at elevated positions with different angles and distances. In addition, their paper lacks a detailed discussion on whether their framework can track and classify all detection hands in a video simultaneously. Real-world surveillance scenarios often contain multiple individuals, and the ability to process all hands in parallel is needed for robust detection in real-time systems. However, their methodology does not specify if the framework processes hands in a sequential or parallel manner, which raises concerns about its applicability in crowded environments in a real-time manner. In research [11], a proposed system used deep neural networks for real-time "Signal for Help recognition", achieving 98.79% accuracy. Their system consisted of hand landmark extraction for some static gestures using the MediaPipe Hands framework for one hand only, which would lead to the inability to apply it in real scenarios, followed by classification using a custom CNN architecture. The testing was at short distance with well-defined hand regions with static gestures and constant light conditions, and the implementation suffered from major limitations related with their system that was work on one hand only and the limited dataset that included reliance on laptop device camera placed close to hands, failing to address the challenges in the surveillance camera context where detection distances normally exceed several meters.

In research [12], authors developed CNN and TensorFlow models for "Signal for Help" classification based on a dataset containing 112 images that was later extended to 2,352 images using image augmentation techniques. Their method achieved accuracies of 87.5% without data augmentation and 100% with data augmentation. Their dataset suffered from a small size, the adopted approach addressed only one specific motion, and the two-move "Signal for Help" gesture was caught at close range by phone cameras. These settings are quite different from systems in which surveillance is conducted.

While surveillance-specific applications have had limited success, traditional hand gesture recognition has proven more effective in controlled environments.

In research [18], a CNN-based system has relied on preprocessing methods such as bicubic interpolation for image resizing and a transition from 3-channel RGB to grayscale, followed by Roberts edge detection to extract the boundaries of an object. In this case, recognition was achieved through the classification of a still image of twenty-four letters of the ASL alphabet, resulting in an exceptional 99.3% accurate classification and a high precision of 99% with a loss lower than 0.0002.

Though it was demonstrated that the system was effective in real-time, only a small dataset of 240 images was used for the experiments. The experiments ran under direct control in laboratory conditions, with the target being in close proximity, usually 2–5 meters away from the camera. Conversely, a distanced surveillance system would have to deal with the inability of effective background subtraction.

In contrast, most of the hand-tracking experiments are limited to controlled environments, use key-point hand localization methods, and are reliant on close-up images with poor recognition at longer distances and under different lighting conditions [19].

On the other hand, a framework of MediaPipe Hands is capable of providing real-time hand-tracking models in real-time on one's device. One of the other developments in this field is the frames per second evaluation, or localization accuracy, although, in most cases, this parameter was evaluated across short distances below 4 meters. However, as demonstrated in Section 2, MediaPipe Hands has had accuracy degradation problems at more extended distances, disproving the authors' findings.

Meanwhile, YOLO-based pose estimation methods demonstrated significant improvements, allowing accurate frames per second under varying distances.

Another aspect of research related to security applications is the interaction of gesture recognition systems with alert mechanisms. In research [11], an email-based notification system using the Simple Mail Transfer Protocol (SMTP) protocol for sending distress signal notifications. When certain hand gestures that suggest trouble are detected, the system sends predetermined assistance messages to emergency contacts. The notification mechanism through the use of emails is simply not applicable for real emergencies.

While progress has been made in terms of components of hand gesture recognition systems, the integrated solutions for surveillance applications still face a lot of challenges. Most existing studies have been limited to controlled experimental settings, short-range gesture recognition, and restricted sets of gestures. However, the creation of robust systems that can function successfully in a variety of distances, environmental settings, and gesture types remains an active area of research, especially for applications that specialize in security, like the detection of distress signals.

The review of the existing literature indicates that current approaches to hand gesture recognition in surveillance application scenarios exhibit four key limitations:

- Limitations of the dataset: There are no standard datasets related to this field of "Signal for Help", just some trials based on close-range captures in controlled settings that do not reflect the environmental diversity found in actual surveillance operations [10–12].
- Limited detection range: Most paradigms show significant performance drop-off beyond about 3–4 meters, which is regarded as well below typical operational parameters of such surveillance systems [5, 10, 13, 16].
- Finally, there are several limitations imposed on the processing pipeline:

First, as indicated in Section 2 of the literature review, no works considered the ability of the framework to classify each hand at the same time. This prevents developing end-to-end solutions for security as the pipeline cannot achieve integration between detection, tracking, classification, and alert generation. Examples of imposed limits include the MediaPipe Hands framework, which allows detecting only two hands in one frame, making it unsuitable for the multi-person surveillance case. The proposed system solves these limitations by adopting end-to-end architecture, expanding detection ranges, ensuring processing in multi-threads, introducing temporal classification, and allowing for alert generation, making the designed system a holistic one for the surveillance domain.

Table 1 summarizes the most relevant related works with an explanation of the main contribution, performance, and limitation.

3. Methodology

This section presents the methodological framework implemented for hand gesture recognition in surveillance environments to detect distress signals associated with abduction scenarios. The proposed system comprises three phases: dataset creation, building and training an optimized LSTM model, and deploying the optimized model with the real-time notification system, as illustrated in Fig. 1. Each phase aims to resolve the designated set of technical challenges to finalize the development of the proposed abduction detection system as follows:

The first phase entails the recording of synthesized videos to replicate abduction patterns and the implementation of a hand-tracking algorithm.

Table 1. Summary of related work.

Ref	Year	Method	Dataset	Accuracy/Results	Key Limitations
[14]	2020	Kalman filter + Microsoft Kinect	Kinect depth-camera dataset (lab)	Stable trajectory tracking in controlled settings	Performance declines >2 meters; constrained lab conditions
[15]	2021	CNN for ASL recognition	Four ASL datasets	99.9–100% accuracy	Controlled capture; short-range (<2 meters)
[10]	2021	CNN + TensorFlow	112 images via phone-camera (augmented to 2,352)	87.5% without augmentation; 100% with augmentation	Small dataset, close-range
[13]	2021	KLT (modified) + Camshift tracking	Indoor custom dataset	>99% recognition on typical gestures	Indoor, close-range only; limited generalizability
[11]	2022	MediaPipe + Deep Neural Network (DNN)	Laptop-camera captures	98.79% accuracy	Close-range, single-hand evaluation; controlled lighting
[17]	2022	Modified LSTM	Leap motion of Chinese Sign Language dataset	72.3–89.5% accuracy	close-range only <2meters, sensors needed
[12]	2023	Dual-CNN	Mobile phone-camera captures	91.25% @16 FPS	<4m, unclear multi-hand processing
[9]	2023	CNN-BiLSTM with attention (data glove)	Wearable (data-glove) datasets	95.05%	Requires wearable sensors, not suitable for non-contact surveillance
[20]	2024	YOLOv8-PoseBoost (pose/keypoint detection)	General object/pose datasets	Improved pose detection at extended ranges	Not directly applied to hand-gesture recognition tasks

Before describing it, it is butter to state that Section 3.1 and Section 3.2 below explain the design of the Abductees-Rescue dataset. As far as the author knowledge, there is no other paper where there was an effort to develop a dataset for hand gesture recognition for surveillance, conventional hand signalization, and particularly abduction. Based on Section 3.1 and Section 3.2, this research had to develop the Abductees-Rescue dataset.

3.1. Designing the rescue-abductees dataset

The review of existing literature revealed a significant gap in available datasets for hand gesture recognition in surveillance contexts, particularly for abduction-related signals. While previous studies developed limited datasets within the range of their research, these collections suffer from several critical limitations: they primarily consist of close-range captures (within 1–2 meters), utilize mobile or laptop cameras rather than surveillance equipment, feature constrained environmental conditions, and contain insufficient sample diversity. The limitations of the design of such datasets make them inadequate for training robust models capable of operating in authentic surveillance environments where cameras are typically mounted at elevated positions and must function across extended distances with variable lighting conditions. To address these deficiencies, this paper developed the Abductees-Rescue dataset, specifically designed to reflect real-world surveillance scenarios. The following sections detail the methodology employed in creating the dataset, including the data acquisition process, hand tracking algorithm implementation, and feature extraction procedures.

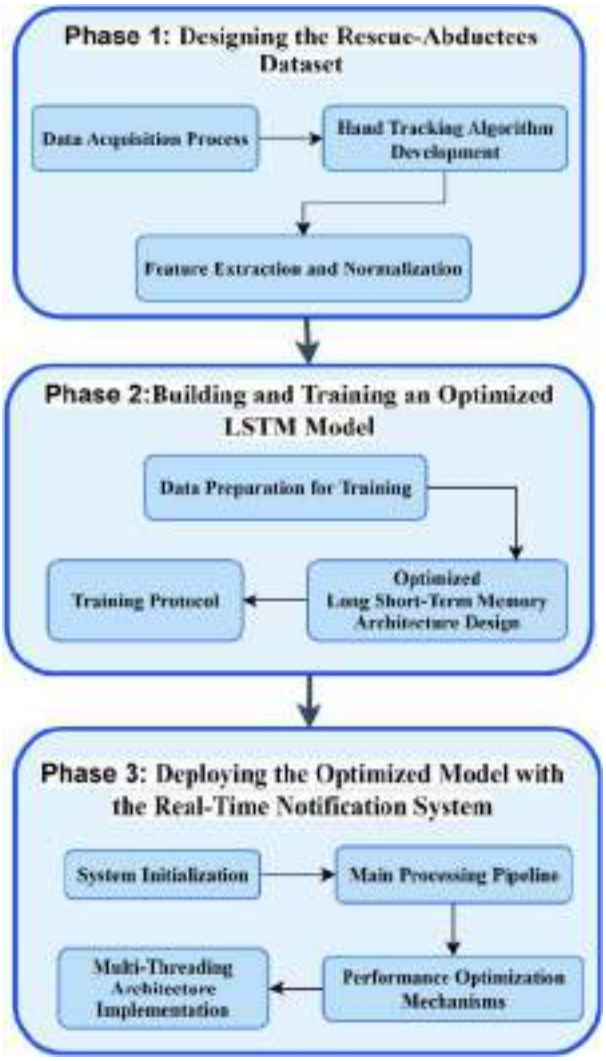

Fig. 1. Block diagram illustrates the three phases of the proposed system.

3.1.1. Data acquisition process

The Abductees-Rescue dataset was created to address the deficiency in surveillance-oriented hand gesture recognition datasets specifically tailored for abduction situations. The acquisition process utilized surveillance cameras (EZVIZ 5MP) mounted at a height of 3 meters to capture authentic surveillance configurations. The approach of creating this dataset deviates significantly from conventional datasets that typically employ mobile devices or laptop cameras positioned at eye level. The recording protocol incorporated the following parameters:

- **Distance variation**: Subjects performed hand gestures at distances more than 10 meters from the camera, establishing a spatial distribution that reflects surveillance operational parameters at different distances.
- **Environmental diversity**: Videos were captured under varied illumination conditions (daytime natural lighting, artificial indoor lighting, and nighttime conditions) across

Fig. 2. Illustrates applying the developed hand tracking algorithm.

multiple locations (indoor environments, outdoor urban settings, and transitional spaces).

- **Hand gesture types**: Two categories were established, abduction-related hand gestures based on the "Signal for Help" pattern (labeled as "1") and normal hand movements representing typical gestural communication (labeled as "0").
- **Subject diversity**: Volunteers aged between 16 to 50 years with varying physical characteristics performed the gestures to ensure representational diversity.

The dataset comprises 9,111 samples (4,545 normal hand gestures and 4,566 abduction-related hand gestures), with each sample consisting of a sequential series of 45 frames. This sample size enables robust model training, while the balanced class distribution mitigates potential classification bias.

3.1.2. Hand tracking algorithm development

A custom tracking algorithm based on YOLOv8 pose estimation was developed to detect and extract hand regions from surveillance footage, as shown in Fig. 2. The approach used in this development allows reliable hand tracking at extended distances, addressing a significant limitation of existing hand detection systems.

YOLOv8 pose estimation provides two important capabilities; first, it maintains robust body key point detection across extended distances up to 10 meters, and second, it extracts shoulder and hip keypoints that enable proportional calculation of adaptive bounding boxes around hand regions. This adaptive scaling ensures that hand regions are captured with consistent dimensions regardless of subject distance from the camera, creating standardized 300 $\times$ 300 pixel regions suitable for subsequent MediaPipe Hands processing. The YOLOv8n variant was specifically selected for its optimal balance between detection

Fig. 3. Keypoints that extracted through the implementation of YOLOv8n pose estimation.

accuracy and computational efficiency, achieving real-time performance at 15 FPS while maintaining the precision necessary for reliable key point extraction. This architectural decision effectively extends the operational range of hand gesture recognition systems from the 1 to 4 meters constraint of previous approaches to the 10 meter range required for practical surveillance deployments.

In (a), (b), and (c) showing how the adaptive bounding box applying on every hand detected (here showing around one hand only for easier illustration), while (d), (e), and (f) showing the output of applying this algorithm as a new video for each hand detected.

The developed algorithm implements four key components:

- **Pose estimation initialization**: The YOLOv8 pose estimation model is initialized with a confidence threshold (α) set to 0.5 for key point detection.
- **Key point extraction**: From the detected pose key points ($K = k_1, k_2 \ldots k_n$), the positions of wrists (keypoints 9 and 10), shoulders (keypoint 5), and hips (keypoint 11) are extracted as shown in Fig. 3.
- **Adaptive bounding box generation**: For each detected hand, a fundamental unit size (chunk) is calculated as:

$$\text{Chunk} = \text{d\_distance}/4 \tag{1}$$

where d_distance represents the distance between the shoulder and hip key points.

The bounding box dimensions are then computed as:

$$\text{Width} = \text{height} = \text{chunk} \times C \tag{2}$$

Fig. 4. Illustrates the tracked hand after using the MediaPipe Hands framework similar to the Body pose illustration. As shown in the above figure, this process involved.

In which C is a multiplication factor that determines the resulting box size.

– **Hand-to-hand distance measurement**: The algorithm maintains the identity of the hands seen across frames, and the difference between the detected and the tracked position with the previous frame:

$$D\left(h,t\right)=\sqrt{\left(x_h-x_t\right)^2+\left(y_h-y\_t\right)^2} \tag{3}$$

where (x_h, y_h) and (x_t, y_t) are coordinates of hand detection in the current region and tracked region from the previous frame respectively. Special attention is required when distance D(h, t) $<$ T, where T is the maximum threshold of allowable differences, empirically calculated at 100 pixels.

Finally, such an algorithm allows the hands to be tracked throughout the dynamic range of all the frame variance, serving as a basis for further feature extraction and classification:

3.1.3. Feature extraction and normalization

In this step, 3D landmarks were extracted for the 21 three-dimensional landmarks based on the hands and frame per second. Followed by normalization to avoid biases during the model training, as shown in Fig. 4.

Feature extraction and normalization use the following steps:

– **Spatial standardization**: Crops and stretches the tracked/domineering hand box to 300×300 pixels.
– **Temporal standardization**: Every hand gesture instance is composed of precisely 45 discrete frames spread across a 15 FPS grid, modeling a replication uniform.
– **Landmark extraction**: The facts about the hands are processed by the API to extract tangible data through the human actions.

Table 2 illustrates this dataset before normalization. Each sample is the output of applying the MediaPipe framework to one video. Each video is of 45 hand frames.

– **Landmark normalization**: A two-step normalization process is conducted to address hand variations in terms of size, position, and orientation.

a) **Calculation of the relative coordinates w.r.t. the wrist joint**: The first landmark point of each frame, the wrist joint's coordinates, are treated as the reference origin. Let absolute coordinates of all 21 landmarks be denoted as

Table 2. Samples of the abductees_rescue dataset.

Sample	Frame	Landmark 1			...	Landmark 21			CLASS LABEL
		X_1	Y_1	Z_1	...	X_21	Y_21	Z_21	
1	1	0.519722	0.595295	2.52E-07	...	0.50809	0.747957	0.0241830	0
	...	...	...	...	...	...	...	...	
	45	0.825835	0.148248	5.06E-08	...	0.826212	0.06206	0.0042430	
2	1	0.508307	0.519856	5.03E-07	...	0.826212	0.06206	0.0042431	1
	...	...	...	...	...	...	...	...	
	45	0.474766	0.436437	1.42E-08	...	0.464101	0.833784	0.0457061	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
9111	1 to 45	X_1	Y_2	Z_3	...	X_21	Y_21	Z_21	0/1

frame_array. Then, a translation is applied as follows:

$$\text{relative\_frame}[i] = \text{frame\_array}[i] - -\text{wrist\_position} \quad (4)$$

Where i is ranged from 0 to 20, denoting each landmark point, and each landmark is presented in three-dimensional coordinates. This translation transfers the hand representation from frame-centric to hand-centric coordinates, where the wrist becomes a new origin. This is a critical step to normalize hand representation over video frames because recognition of hand signs should be based on the relative configuration of the hand landmarks rather than their absolute position anywhere in the video frame.

b) **Scale the relative coordinates to be within [-1, 1] as follows**:

$$\text{normalized\_frame} = \text{relative\_frame}/\text{max\_val} \quad (5)$$

Where max_val is the maximum absolute value among all coordinates.

This normalization procedure is essential for reducing the model's dependency on different hand sizes, different subjects, and different video recording conditions by converting absolute positional information of hands to relative configuration. Table 3 below shows a sample from the Abductees-Rescue dataset prior to normalization.

The output of this normalization technique is illustrated in Table 4:

3.2. Building and training an optimized long short-term memory model

After creating the Abductees-Rescue dataset, the model consists of automatically classifying hand markers' temporal nature from landmark sequences on a sequence of frames within a video. This next section describes how a perfected LSTM network was engineered to distinguish between regular hand gestures and those expressing distress in a monitoring scenario. While the focus was on the classification's precision, the formulation had to be lightweight enough for real-time processing.

3.2.1. Data preparation for training

The preparation process is as listed:

– **Data Splitting**: The whole dataset was stratified into 30% test ratios and 70% train ratios. This was performed to ensure the same class' uniformity was portrayed.

Table 3. Dataset before normalization.

Before Normalizing One Sample of 3D Hand Landmarks of One Frame			
Points of Landmarks	**X**	**Y**	**Z**
Point 1 (Wrist)	0.5034	0.5896	0.0000
Point 2	0.4385	0.6203	-0.0647
Point 3	0.4156	0.6957	-0.0847
Point 4	0.4048	0.7675	-0.0892
Point 5	0.3904	0.8256	-0.0883
Point 6	0.5194	0.7429	-0.0716
Point 7	0.4994	0.8460	-0.0910
Point 8	0.4581	0.8893	-0.0986
Point 9	0.4205	0.9103	-0.0987
Point 10	0.5525	0.7471	-0.0367
Point 11	0.5238	0.8482	-0.0545
Point 12	0.4749	0.8890	-0.0589
Point 13	0.4305	0.9064	-0.0580
Point 14	0.5626	0.7460	-0.0052
Point 15	0.5296	0.8305	-0.0205
Point 16	0.4836	0.8638	-0.0284
Point 17	0.4418	0.8759	-0.0298
Point 18	0.5584	0.7430	0.0219
Point 19	0.5322	0.8126	0.0068
Point 20	0.4940	0.8423	-0.0018
Point 21	0.4587	0.8549	-0.0033

– **Sequence Preparation**: Since the sequence is flat and doesn't require 3D coordinates, the input frame/intelligence has a sequence measure of 45 frames of 21 normalized landmark points apiece flattened, totaling 63 flattened frames. Therefore, the input tensor's shape is structured from 45 frames to 63 flattened elements. Fig. 7 represents the entire optimized LSTM architecture.
– **Input shape configuration**: the input is formed as 45 temporals, and 63 spatials for each hand sequence.

3.2.2. Optimized long short-term memory architecture design

A hierarchical LSTM network takes the form of how the video behaves in a sequence, particularly the temporal perception of hand dynamics, as Fig. 5 shows. The structural design consists of optimized methods such as:

– **Hierarchically Structured LSTMs**: A 2-fold LSTM model integrated 64 unit elements each.
– **Layer-wise Regularization**: The two LSTM layers had L2 regularization of 0.02 and 0.01 to prevent the model from overfitting.

Table 4. The output of applying the normalization technique on twenty-one 3D hand landmarks of one sample.

After Normalizing the Previous Sample			
Points of Landmarks	**X**	**Y**	**Z**
Point 1 (Wrist)	0.0000	0.0000	0.0000
Point 2	-0.2024	0.0957	-0.6557
Point 3	-0.2738	0.3306	-0.8581
Point 4	-0.3074	0.5540	-0.9040
Point 5	-0.3520	0.7360	-0.8949
Point 6	0.0499	0.4778	-0.7257
Point 7	-0.0125	0.7998	-0.9219
Point 8	-0.1417	0.9545	-0.9990
Point 9	-0.2588	1.0000	-1.0000
Point 10	0.1532	0.4015	-0.3719
Point 11	0.0657	0.8058	-0.5522
Point 12	-0.0890	0.9336	-0.5968
Point 13	-0.2275	0.9879	-0.5877
Point 14	0.1845	0.4879	-0.0527
Point 15	0.0817	0.7505	-0.2077
Point 16	-0.0618	0.8553	-0.2878
Point 17	-0.1921	0.8969	-0.3020
Point 18	0.1716	0.4849	0.2218
Point 19	0.0899	0.6929	0.0689
Point 20	-0.0294	0.7881	-0.0182
Point 21	-0.1395	0.8279	-0.0334

– **Batch Normalization and Dropouts**: Each LSTM layer structured batch normalization and dropping units of 50%.
– **Output Configuration**: A unit of sigmoid activation structured to distinguish between a normal hand gesture and an abducted gesture.

3.2.3. Training protocol

Four additional selections were made to boost the model's performance and bring about generalization. These are:

– **Batch Size Optimization**: A batch of 32 optimized the computation and allowed the model to evaluate coefficients efficiently.
– **Early Stopping Mechanism**: It terminated the process after 300 cycles of decreasing loss in the testing set, with a tolerance of 10 units, to prevent overfitting and save the best coefficients. Moreover, the mechanism saves the model's coefficients with the least loss in the testing set.
– **Model Checkpointing Increment**: A checkpoint was incremented by a factor of one.

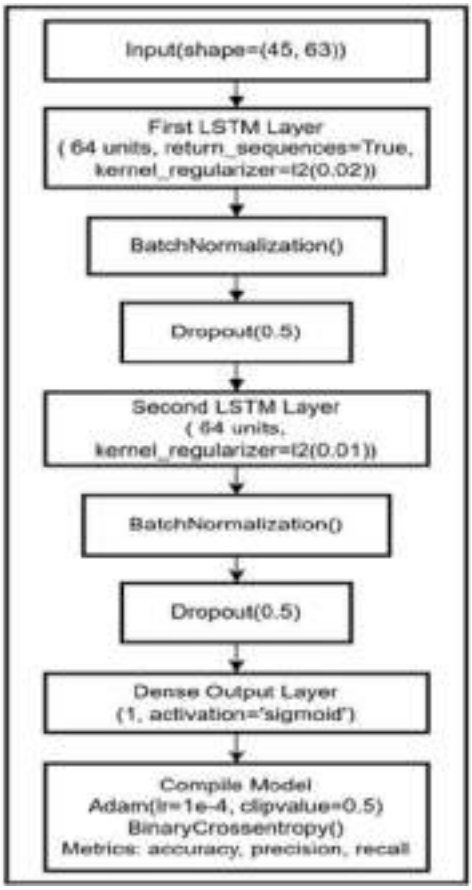

Fig. 5. Block diagram of building an optimized LSTM model.

– **Adaptive Learning Rate Reduction**: The strategy reduced the learning rate by a factor of 0.5.

3.3. Deploying the optimized model with the real-time notification system

A processing pipeline of multithreads is formed that detects and classifies hand gestures in real-time with the power of the computer and the webcam. The optimized model is used and integrated into the pipeline that extracts frames, performs 3D wrist location detection via Yolov8, records 2D wrist locations, classifies the gestures performed by the hands, and recognizes individual hands on a surveillance stream. The flowchart structure and information exchange are shown in the figure below. Mechanisms of performance optimization include Frame Skipping, Resolution Scaling, Async Processing, and Hand Tracker removal. Fig. 6 illustrates the flow work architecture of this system.

3.3.1. System initialization

The inference framework begins by initializing the required components:

– **Environment configuration**: Setup of computational resources, including GPU memory allocation, threading parameters, and model loading.

- **Model initialization**: including the YOLOv8 pose estimation model for detecting body keypoints, the MediaPipe Hands model pool for parallel hand landmark extraction, and a pre-trained LSTM model for gesture classification.
- **Camera connection**: Establishment of connection to surveillance camera (EZVIZ 5MP) with video parameters (15 FPS, resolution: 2880 $\times$ 1620 pixels).
- **Thread pool creation**: Initialization of thread pools for parallel processing of multiple hand regions.

3.3.2. Main processing pipeline

The real-time processing of the hand gesture recognition is done via a multi-stage pipeline as follows:

- **Frame acquisition and preprocessing**: Including the continuous reading of frames from the camera at 15 FPS and applying YOLOv8 pose estimation to detect body keypoints.
- **Hand detection and tracking**: Including extraction of hand positions (wrists) from pose estimation results, implementation of adaptive bounding box generation based on body proportions, application of identity tracking algorithm to maintain consistent hand identification across frames, and implementation of velocity-based prediction for occlusion handling.
- **Multi-threaded hand processing**: Including creation of separate processing threads for each detected hand, parallel processing of multiple hands to maximize computational efficiency, and thread synchronization mechanisms to manage shared resources.
- **Hand landmark extraction and normalization**: Including the MediaPipe Hands application to each hand region, extraction of 21 three-dimensional landmarks per hand, normalization of landmarks relative to the wrist position, and organization of normalized landmarks into fixed-length sequences.
- **Gesture classification**: Including application of the pre-trained LSTM model to each hand sequence, binary classification with a probability threshold (0.5), and temporal consistency verification to reduce false positives.
- **Alert generation and notification**: Including visual marking of detected distress signals with red bounding boxes, capture and saving of frames containing detected signals, and transmission of alerts through Telegram Bot API with photographic evidence.

3.3.3. Performance optimization mechanisms

Several optimization strategies were implemented to ensure real-time performance:

- **Frame skipping**: Processing of every Nth frame (where N is configurable) to reduce computational load while maintaining detection capabilities.
- **Resolution scaling**: Reduction of input frame resolution for pose estimation while maintaining original resolution for visualization and alert generation.
- **Model pooling**: Creation of multiple instances of detection models to enable parallel processing without resource contention.
- **Asynchronous processing**: The asynchronous alert notification system operates through a dedicated execution thread pool separate from the main gesture processing pipeline. When the LSTM classifier detects an abduction-related gesture (probability $\geq$ 0.5), the system immediately captures the current frame, annotates it with a red bounding box around the detected hand region, generates a unique alert identifier with a timestamp, and submits the notification task to the alert queue without blocking the main processing thread. The alert worker thread retrieves queued notifications, formats the photographic evidence with metadata (detection time, confidence score, processing

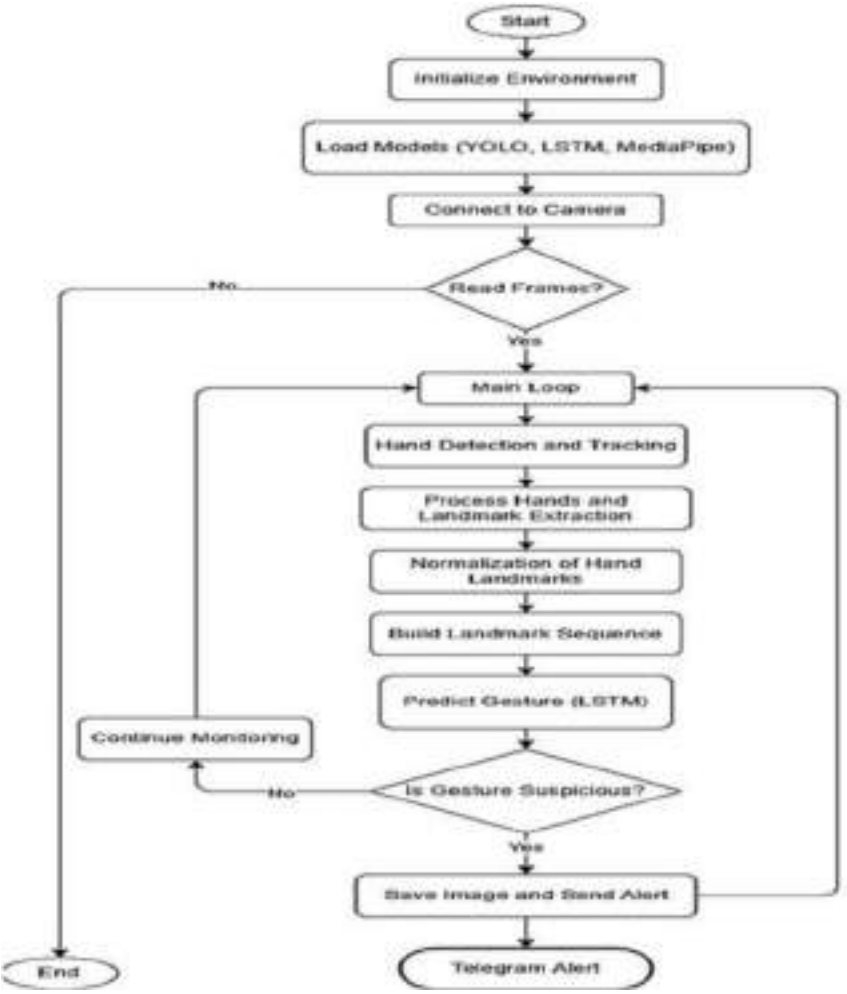

Fig. 6. Flowchart of hand gesture-related Abduction case recognition system.

duration), and transmits the package through the Telegram Bot API. This architectural separation ensures that network latency or API response delays do not impact the continuous 15 FPS processing rate of the main surveillance pipeline. The system logs all alert transmission attempts with success/failure status and delivery timestamps for audit trail purposes.

– **Resource management**: Periodic cleanup of inactive hand trackers and detection history to maintain memory efficiency.

3.3.4. Multi-threading architecture implementation

The multi-threaded processing architecture instantiates a ThreadPoolExecutor with a fixed pool size of 6 worker threads, proactively chosen based on two crucial considerations:

– The maximal expected number of hands in the surveillance frame
– The number of available CPU logical cores

The latter amounts to 12 in the deployment system's Intel Core i7-9750H processor, with 6 of those cores dedicated to the gesture processing tasks and 2 remaining cores for

stable system performance and background processes, which eliminates the chance of CPU saturation.

Each worker thread is responsible for a distinct portion of the processing pipeline and includes:

- Receiving the hand region coordinates from the central detection loop
- Applying the MediaPipe Hands routine to extract the landmarks in the cropped 300×300 pixel area
- Normalizing the landmarks wrist-relative and scaling their absolute coordinates
- Aggregating the normalized landmarks into 45-frame temporal sequences
- Passing the completed sequences to the LSTM classifier for the gesture classification task

The data flow between the main detection loop and worker threads is organized via thread-safe queue structures, with each independently tracked hand assigned a Universally Unique Identifier to ensure consistent hand tracking throughout consecutive frames within session duration.

3.4. Implementation environment

The system was implemented and tested on a high-performance computing platform equipped with an NVIDIA Quadro T2000 GPU, with a total memory of 20GB, where 4GB is dedicated display memory and 16GB is shared memory.

The platform comprises an Intel Core i7-9750H processor with a CPU running at 2.60GHz, 32GB of system RAM, and is operating in a Windows 11 Pro 64-bit environment.

This high-performance configuration is specialized to provide enough computational resources required for real-time utilization processing due to multiple video streams used in capturing and real-time parallel analysis processing of the hand gestures.

The system software is built in Python programming language along with the deep learning frameworks for control of neural network operations and OpenCV for image-intensive processing tasks.

A 5MP surveillance camera, electronic EZVIZ at 15 FPS, mounted at a height of 3 meters, was employed for the video capture and processing to emulate an authentic surveillance setup configuration.

The alert transmission is achieved through the implementation of the Telegram Bot API for the immediate delivery of the alert with evidence photographs.

The major concerns during the implementational design comprised achieving a real-time facility, ensuring system stability and reliability in detecting and recognizing the hand in various environmental conditions and running distances in real-time.

4. Performance analysis

This section presents the performance analysis of the proposed hand gesture recognition system for abduction distress gesture detection from the surveillance system's perspective. Performance analysis of hand detection and tracking, gesture classification, and the system's performance in different environmental conditions is given below.

4.1. Hand detection and tracking performance

Based on the experimental evaluation of the proposed hand gesture recognition system, it is evident that there was a considerable improvement in performance ranking from

Fig. 7. Applying the developed adaptive bounding box algorithm.

detection to tracking achieved in various environments. Initially, this performance improvement may arise from the incorporation of the hand detection and tracking technique, which leverages the YOLOv8 pose estimation model.

Developing the adaptive bounding box for the technique will dynamically adjust to hand distance proximities and body distances considering hip-to-shoulder points. The adaptive bounding box using the YOLOv8 pose estimation algorithm implementation addresses the limitations in existing hand detection and tracking frameworks, especially at extended ranges such as 10 meters.The algorithm demonstrates performance in maintaining consistent hand tracking within the coverage distance range of the camera, while effectively handling occlusions and temporary detection failures. It maintains hand region capture consistency even with varying distances from the camera up to 10 meters. Fig. 7 illustrates

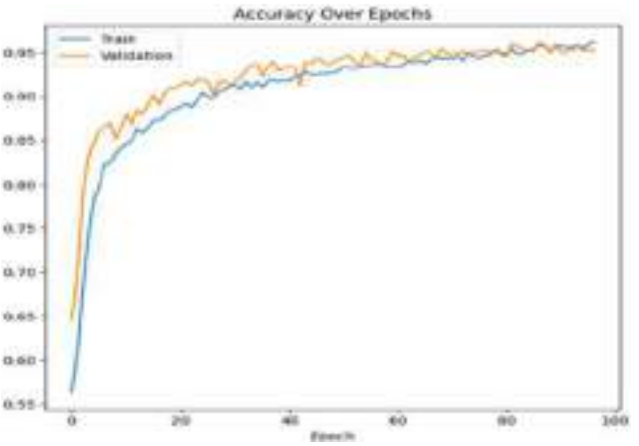

Fig. 8. Training validation accuracy progression of the model.

how the size of the adaptive bounding boxes grows based on the distance between the people and the surveillance camera.

Frame (a): the tracked hands at 10 meters distance, (b): the tracked hands at 8 meters distance, (c): the tracked hands at 5 meters distance, and (c): the tracked hands at 0.5 meters distance.

4.2. Hand gesture classification accuracy

Table 5 summarizes the performance of the optimized LSTM architecture for temporal logs' classification; differentiating ordinary hand gestures and abduction signals is presented. The model recorded averages of 96.12% for accuracy, 95.86% for precision, and 96.42% for recall. The observation of the generality of the measurement indicates that the classifier does not have any systematic bias to classify any given gesture, and hence, it would operate efficiently as a surveillance system. The optimization of the architecture defined convergence to be after 97 epochs based on the early stop mechanism. The marginal metric between the training and validation datasets, at about 0.75% for the accuracy measurement, indicates average competitiveness; hence, the clinical model did not overfit as shown in Fig. 8. During this phase, the increase in the validation loss, which was approximately 0.0387, for the training is within the acceptable margin for any RNN applied to temporal classification, as represented in Fig. 9.

The confusion matrix indicates a near symmetry in the distribution of the misclassified orders for the two classes: 49 false negatives and 57 false positives, as presented in Table 5. There is affirmation that the model has equally well-learned discriminant measures for the two categories. The F1-scores defined by both precision and recall measurements were 0.96 for both classes, affirming a balanced model for both precision and recall.

4.3. Comparative performance analysis

The performance measure of the optimized LSTM model was 96.12% on the abduction-Rescue dataset. The dataset was designed for gesture recognition as a classification problem in the context of surveillance. This achievement is impressive because the performance on the dataset involved a video camera covering more than 10-meter distance, fluctuation of the environmental and illumination area, and having real surveillance setups. In addition,

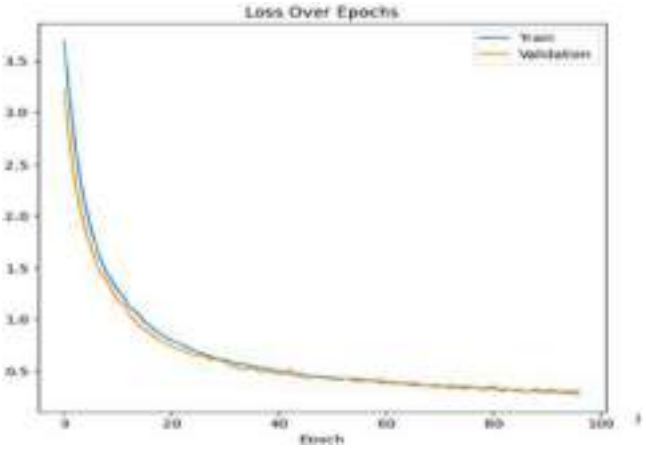

Fig. 9. Training and validation loss reduction through the optimization process.

Table 5. Illustrates the confusion matrix of training the LSTM model.

Confusion Matrix	Predict Normal	Predict Abduction-related gestures
Normal hand gestures	1307 (TN)	57 (FP)
Abduction-related hand gestures	49 (FN)	1321 (TP)

the suggested approach verifies an impressive achievement toward the development of surveillance systems. Table 6 presents a comprehensive comparison of the proposed system with the related works, highlighting key performance metrics and operational limitations. It should be noted that the comparison still mainly theoretical, because of the different datasets used, experimental setting, and environmental conditions, in spite of the similarity in the principal gesture distinguishing approaches. So, the results underline the efficiency of the proposed system within its particular implementation situation rather than proposing straightforward numerical advantage over previous approaches.

4.4. System operational capabilities

The real-time processing capability represents a key achievement of the implemented system, where the performance testing demonstrated the ability to process surveillance footage at the native camera frame rate (15 FPS) without significant latency. The multi-threaded architecture achieves high computational resource utilization to enable simultaneous tracking and classification of multiple hands, decoupling the fundamental limitation of existing frameworks, such as MediaPipe Hands, that limit detection to 2 hands per frame. The system's hand detection and tracking feature works at extended range, countering its current limitation in existing approaches, i.e., MediaPipe Hands. Most current conventional hand-tracking frameworks quickly degrade in performance beyond 3–4 meters away.

The developed system enabled the detection and tracking of hands at a range of more than 10 meters away from the camera. This extended distance is the range at which a typical surveillance system is installed and, therefore, the system can be fit within limited infrastructure.

Table 6. Performance comparison with related works.

Ref	Method	Dataset	Testing Accuracy	Key Limitations
[10]	Dual-CNN	Mobile phone-camera captures	91.25%	<4m, unclear multi-hand processing.
[11]	MediaPipe + DNN	Laptop-camera captures	98.79%	Close-range, single-hand evaluation; controlled lighting.
[12]	CNN + TensorFlow	112 images via phone-camera (augmented to 2,352)	87.5% without augmentation; 100% with augmentation	Small dataset, close-range
The Proposed System	LSTM + YOLOv8 + Adaptive bounding box + MediaPipe Hands framework.	9,111 samples extracted from videos of the tracked hands via the MediaPipe framework	96.12%	Requires gesture awareness. People with disabilities, such as those who are missing some fingers, were not taken into consideration in this work.

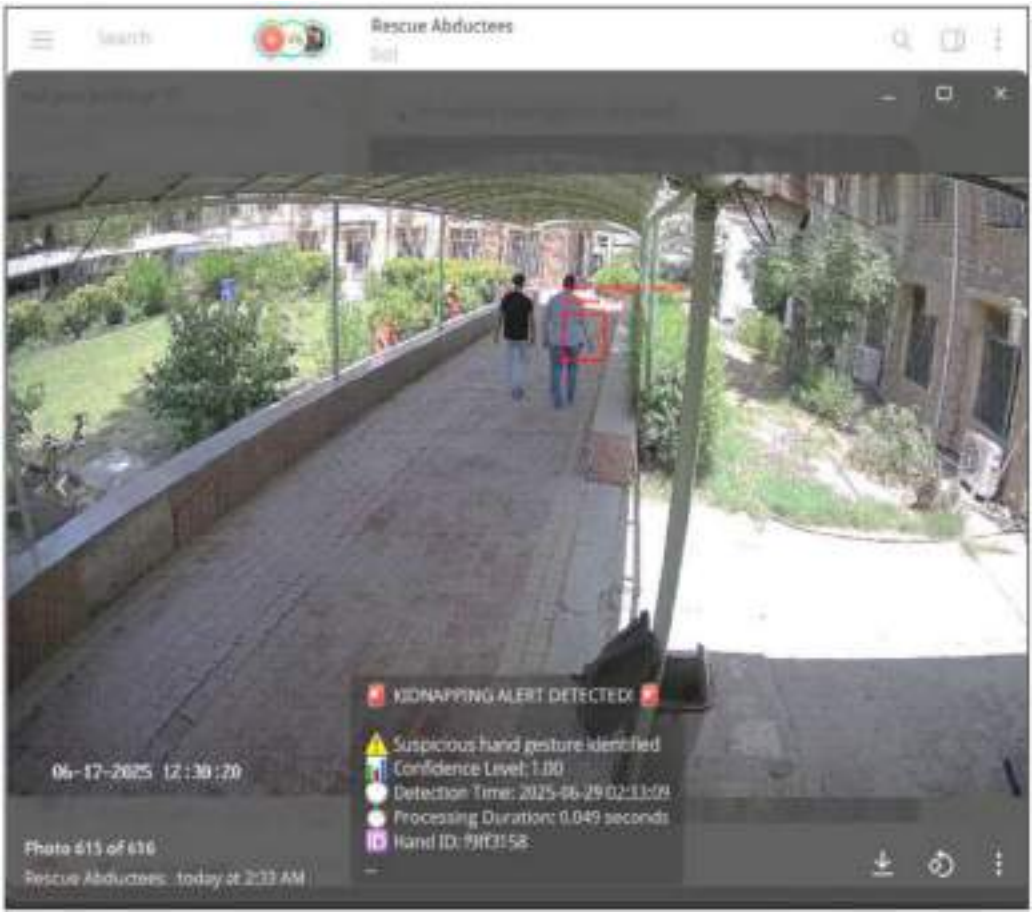

Fig. 10. Real-time notification system with processing report.

Another feature of the system is that it can send an immediate notification via a call from photographic evidence and a brief description. This feature counters the existing alert email delivery models that have some flaws in delaying delivery without some urgent attention mechanism. The call gives room for immediate action in a potential abduction situation, which is a factor in successful intervention.

The average processing time of detecting the abduction-related hand gesture is 0.049 seconds, while the average total time until the alarm arrives in the Telegram bot is 1.839 seconds from gesture detection up to Telegram arrival.

Fig. 10 shows the real-time notification system on determine how an abduction occurs with the message.

4.5. Limitations

The proposed system has certain limitations and is not reliable in some operating conditions from the technical perspective. While the performance of the system deteriorates for distances over 10 meters owing to possible environmental issues and occasional lighting conditions that could not be explored extensively, another limitation is the concentration of the dataset on one specific type of standardized “Signal for Help” gesture, which requires enormous educational endeavors from the side of the humanitarian agencies to allow the potential victims to know and efficiently use the distress signals.

5. Conclusions and future work

This paper has presented a new real-time hand gesture recognition system that aims to identify applications expressing the signals of distress connected to the abduction situations within the programming environments. The limitations in the existing works have been addressed through a blend of long-term assistance with interaction and duration categorization and warning formation. Utilizing the created YOLOv8 pose assessment adaptive bounding boxes, the hand monitoring conducted was excellent within a range spanning more than 10 meters, far beyond the 3 to 4 meters range restrictions of existing systems. Operating at the 96.12 percent degree of categorization precision across multiple environmental conditions, the optimized LSTM design proves robust in separating abduction-related signals and natural hand gestures.

The generation of the Abductees-Rescue data is a significant contribution to the paper’s impact to address an existing vacuum within the field. The data creation strategy adopted provides a more appropriate starting point for models that intend to function in real-world programs by enabling what a structure designed for implementation might generalize in light of the deemed datasets appropriate for use in surveyed studies found in the previously included papers, which depend on various different mobile or laptop cameras to read a limited number of particular visuals or images posed at eye level under controlled settings.

The proposed multi-threaded processing pipeline effectively solves the limitation of the MediaPipe Hands framework that could only detect two hands per frame; thus, swiping was done across the frame to extract the 3D hand landmark features supported by simultaneous detection and tracking. Additionally, the multi-threading processing enables the classification process to be performed on thousands of hands appearing in real-time surveillance footage.

The use of the Telegram messaging platform alerts ensures that there is transmission of alerts accompanied by evidence of photos. Therefore, response in case of abduction cases is complemented. In addition, the proposed system is a noteworthy contribution towards

increasing public safety by ensuring that distress signals can be detected in the surveillance system; thus, effective measures can be taken when a potential abduction case is noted.

Future work would involve adding attention mechanisms in the LSTM model to ensure that the most discriminative snippets in the gesture representation are ensured, thus reducing instances of false positives when normal gestures look like distress for a moment. Moreover, the authors plan to collect more samples by moving to different locations to ensure that the datasets have balanced samples from all scenarios.

The authors would like to thank all who contributed to creating our Abductees-Rescue dataset and thanks to Mendeley who helped us with publication.

Acknowledgement

The authors would like to express their sincere gratitude to the volunteers who participated in creating the Abductees-Rescue dataset, without whom this paper would not have been possible.

Disclosure of conflict of interest

The author declares no conflict of interest.

Ethical approval

This paper has pictures of human subjects. All participant were informed of the procedure purpose, including the use of there images and recording for research and publication purposes.

Data availability

https://data.mendeley.com/datasets/cgf9z2kbn4/1.

References

1. M. Oudah, A. Al-Naji, and J. Chahl, "Hand gesture recognition based on computer vision: a review of techniques," *Journal of Imaging*, vol. 6, no. 8, Jul. 2020, Art. no 73, 2020, doi: 10.3390/jimaging6080073.
2. Z. N. Razoqi, R. Ogla, and A. M. S. Rahma, "Modern face recognition systems: A review of methods and empirical findings," *Journal of Soft Computing and Computer Applications*, vol. 2, no. 1, Jun 2025, Art. no. 1017, doi: 10.70403/3008-1084.1017.
3. I. Vladova and M. Kuleva, "Signal for help," *Strategies for Policy in Science and Education*, vol. 29, no. 4s, pp. 74–83, 2021, doi: 10.53656/str2021-4s-8-help.
4. L. Guo, Z. Lu, and L. Yao, "Human-machine interaction sensing technology based on hand gesture recognition: A review," *IEEE Transactions on Human-Machine Systems*, vol. 51, no. 4, pp. 300–309, Aug. 2021, doi: 10.1109/THMS.2021.3086003.
5. F. Zhang *et al.*, "Mediapipe hands: On-device real-time hand tracking," 2020, *arXiv:2006.10214*.
6. M. Al-Hammadi *et al.*, "Deep learning-based approach for sign language gesture recognition with efficient hand gesture representation," *IEEE Access*, vol. 8, pp. 192527–192542, 2020, doi: 10.1109/ACCESS.2020.3032140.
7. J. P. Sahoo, A. J. Prakash, P. Pławiak, and S. Samantray, "Real-time hand gesture recognition using fine-tuned convolutional neural network," *Sensors*, vol. 22, no. 3, Jan. 2022, Art. no. 706, doi: 10.3390/s22030706.

8. T. H. Obaida, A. S. Jamil, and N. F. Hassan, "Real-time face detection in digital video-based on Viola-Jones supported by convolutional neural networks," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 3, pp. 3083–3091, Jun. 2022, doi: 10.11591/ijece.v12i3.pp3083-3091.
9. J. Wu, P. Ren, B. Song, R. Zhang, C. Zhao, and X. Zhang, "Data glove-based gesture recognition using CNN-BiLSTM model with attention mechanism," *Plos One*, vol. 18, no. 11, Nov. 2023, Art. no. e0294174, doi: 10.1371/journal.pone.0294174.
10. S. Azimi, C. D. Sio, F. Carlucci, and L. Sterpone, "Fighting for a future free from violence: A framework for real-time detection of "signal for help"," *Intelligent Systems with Applications*, vol. 17, Feb. 2023, Art. no. 200174, doi: 10.1016/j.iswa.2022.200174.
11. N. F. Thejowahyono, M. V. Setiawan, S. B. Handoyo, and A. H. Rangkuti, "Hand gesture recognition as signal for help using deep neural network," *International Journal of Emerging Technology and Advanced Engineering*, vol. 12, no. 2, pp. 37–47, Feb. 2022, doi: 10.46338/ijetae0222_05.
12. G. Elliott, K. Meehan, and J. Hyndman, "Using CNN and tensorflow to recognise 'Signal for Help' hand gestures," in *Proc. 2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conf. (UEMCON)*, New York, NY, USA, pp. 0515–521, doi: 10.1109/UEMCON53757.2021.9666484.
13. S. Saboo and J. Singha, "Vision based two-level hand tracking system for dynamic hand gestures in indoor environment," *Multimedia Tools and Applications,* vol. 80, no. 13, pp. 20579–20598, Mar. 2021, doi: 10.1007/s11042-021-10669-7.
14. L. Yang, M. Wang, and T. Li, "Hand tracking based on the Kinect and Kalman filter," in *Proc. 2020 IEEE 3rd Int. Conf. on Information Systems and Computer Aided Education (ICISCAE)*, Dalian, China, pp. 708–714, doi: 10.1109/ICISCAE51034.2020.9236903.
15. A. Sharma, N. Sharma, Y. Saxena, A. Singh, and D. Sadhya, "Benchmarking deep neural network approaches for Indian Sign Language recognition," *Neural Computing and Applications,* vol. 33, no. 12, pp. 6685–6696, Oct. 2020, doi: 10.1007/s00521-020-05448-8.
16. Y. S. Tan, K. M. Lim, and C. P. Lee, "Hand gesture recognition via enhanced densely connected convolutional neural network," *Expert Systems with Applications*, vol. 175, Aug. 2021, Art. no. 114797, doi: 10.1016/j.eswa.2021.114797.
17. B. Wu, Z. Lu, and C. Yang, "A modified LSTM model for Chinese sign language recognition using leap motion," in *Proc. 2022 IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC)*, Prague, Czech Republic, pp. 1612–1617, doi: 10.1109/SMC53654.2022.9945287.
18. A. A. Abdulhussein and F. A. Raheem, "Hand gesture recognition of static letters American sign language (ASL) using deep learning," *Engineering and Technology Journal*, vol. 38, no. 6, pp. 926–937, Jun. 2020, doi: 10.30684/etj.v38i6A.533.
19. A. A. Fahad, H. J. Hassan, and S. H. Abdullah, "Real-time hand gesture extraction using Python programming language facilities," *Engineering and Technology Journal*, vol. 39, no. 06, pp. 1031–1040, Jun. 2021, doi: 10.30684/etj.v39i6.1619.
20. F. Wang, G. Wang, and B. Lu, "YOLOv8-poseboost: Advancements in multimodal robot pose keypoint detection," *Electronics*, vol. 13, no. 6, Mar. 2024, Art. no. 1046, doi: 10.3390/electronics13061046.