Deep Learning-Based Human Body Posture Recognition and Tracking for Unmanned Aerial Vehicles

Authors:

Min-Fan Ricky Lee, Yen-Chun Chen, Cheng-Yo Tsai

Date Submitted: 2023-02-21

Keywords: activity recognition, deep learning, pose estimation, unmanned aerial vehicles

Abstract:

For many applications (e.g., surveillance and disaster response), situational awareness is essential. In these applications, human body posture recognition in real time plays a crucial role for corresponding response. Traditional posture recognition suffers from accuracy, due to the low robustness against uncertainty. Those uncertainties include variation from the environment (e.g., viewpoint, illumination and occlusion) and the postures (e.g., ambiguous posture and the overlap of multiple people). This paper proposed a drone surveillance system to distinguish human behaviors among violent, normal and help needed based on deep learning approach under the influence of those uncertainties. First, the real-time pose estimation is performed by the OpenPose network, and then the DeepSort algorithm is applied for tracking multi-person. The deep neural network model (YOLO) is trained to recognize each person's postures based on a single frame of joints obtained from OpenPose. Finally, the fuzzy logic is applied to interpret those postures. The trained deep learning model is evaluated via the metrics (accuracy, precision, recall, P-R curve and F1 score). The empirical results show the proposed drone surveillance system can effectively recognize the targeted human behaviors with strong robustness in the presence of uncertainty and operated efficiently with high real-time performance.

Record Type: Published Article

Submitted To: LAPSE (Living Archive for Process Systems Engineering)

Citation (overall record, always the latest version):	LAPSE:2023.0823
Citation (this specific file, latest version):	LAPSE:2023.0823-1
Citation (this specific file, this version):	LAPSE:2023.0823-1v1

DOI of Published Version: https://doi.org/10.3390/pr10112295

processes

MDPI

Article

Deep Learning-Based Human Body Posture Recognition and Tracking for Unmanned Aerial Vehicles

Min-Fan Ricky Lee [1,2,*], Yen-Chun Chen [1] and Cheng-Yo Tsai [1]

[1] Graduate Institute of Automation and Control, National Taiwan University of Science and Technology, Taipei 106335, Taiwan
[2] Center for Cyber-Physical System Innovation, National Taiwan University of Science and Technology, Taipei 106335, Taiwan
* Correspondence: rickylee@mail.ntust.edu.tw

Abstract: For many applications (e.g., surveillance and disaster response), situational awareness is essential. In these applications, human body posture recognition in real time plays a crucial role for corresponding response. Traditional posture recognition suffers from accuracy, due to the low robustness against uncertainty. Those uncertainties include variation from the environment (e.g., viewpoint, illumination and occlusion) and the postures (e.g., ambiguous posture and the overlap of multiple people). This paper proposed a drone surveillance system to distinguish human behaviors among violent, normal and help needed based on deep learning approach under the influence of those uncertainties. First, the real-time pose estimation is performed by the OpenPose network, and then the DeepSort algorithm is applied for tracking multi-person. The deep neural network model (YOLO) is trained to recognize each person's postures based on a single frame of joints obtained from OpenPose. Finally, the fuzzy logic is applied to interpret those postures. The trained deep learning model is evaluated via the metrics (accuracy, precision, recall, P-R curve and F1 score). The empirical results show the proposed drone surveillance system can effectively recognize the targeted human behaviors with strong robustness in the presence of uncertainty and operated efficiently with high real-time performance.

Keywords: activity recognition; deep learning; pose estimation; unmanned aerial vehicles

Citation: Lee, M.-F.R.; Chen, Y.-C.; Tsai, C.-Y. Deep Learning-Based Human Body Posture Recognition and Tracking for Unmanned Aerial Vehicles. *Processes* **2022**, *10*, 2295. https://doi.org/10.3390/pr10112295

Academic Editor: Xiong Luo

Received: 2 August 2022
Accepted: 19 October 2022
Published: 4 November 2022

1. Introduction

In recent years, there has been an increasing interest in the application of Unmanned Aerial Vehicles (UAVs). The high maneuverability allows UAV traversing into harsh environments easily in comparison to humans. Camera is a common sensor on UAV [1–3] and can be used to monitor and recognize human movements and poses [4–6].

For many applications of UAVs, such as surveillance and disaster response, situational awareness is essential. In these applications, one of the main tasks is to find a specific target and respond promptly. Human factors in situational awareness are important topics. Humans can misjudge images or reduce judgment efficiency due to fatigue and inattention. An automated surveillance method can help people interpret video feeds and warn them of people found in the area of interest in real time. Collaboration between humans and agents, such as robotic agents, interweaving human and machine decisions, allow the flexible transfer of control between humans and agents as part of a team, in order to achieve effectiveness that is greater than the sum of the parts of humans and agents' level. "Human-Agent Collectives" is a term used to describe such collaborative works [7].

As one of the most popular subjects in recent years, the application of deep learning is also very suitable for human body pose estimation and posture recognition. Human posture recognition has been widely used in home security, games, entertainment, medical systems and other fields. By contrast, there are still some problems in pose estimation.

First, there is no way of knowing how many people are in the picture that would be checked. Second, human interaction can result in complex spatial interference and make identification difficult. Third, the number of individuals in the picture appears to increase the complexity of the operation, and reduce the efficiency of identification. Two main methods are applied to detect human body posture. One is the top-down method, in which all human bodies are detected first, followed by key points on the human body. The downside is that as the number of individuals in the picture grows, the detection time will grow linearly. The other is the bottom-up method. The joints of all people are detected first, and then matched into the human skeleton. The downside is that the information of global context cannot be used [8].

OpenPose is a library for real-time multi-person keypoint detector [9]. The bottom-up human pose detection and Convolution Neural Network (CNN) was proposed to provide developers with data on the human skeleton from images and videos. OpenPose was the first open-source real-time system for multi-person 2D pose detection, including body, foot, hand, and facial keypoints.

Developers are increasingly focused on designing successful applications based on OpenPose. A real-time 2D human gesture grading system from monocular images based on OpenPose was proposed [10]. The 2D positions of a person's joints and skeleton wireframe of the body was captured to generate motion trajectory for every joint. The similarity metric and scoring formula were proposed for grading the gesture. Other application based on OpenPose include a markerless system for gait analysis [11], a promotion system for home-based squat training [12] and robust 3D skeleton tracking [13].

Simple Online and Realtime Tracking (SORT) is an open source for multiple objects tracking with a focus on simple, effective algorithms [14]. They use geometric cues of objects instead of motion and appearance cues to tackle the occlusion and re-identification challenges simultaneously. A Deep SORT algorithm [15] was proposed to extend the SORT by integrating appearance information. The experiment shows that objects track can be tracked through longer periods of occlusions and the number of identity switches can be reduced effectively. It achieved overall competitive performance at high frame rates.

The You Only Look Once (YOLO) is an open-source library for the object detection [16]. The regression approach was proposed for object detection instead of classification. The single neural network predicted spatially separated bounding boxes and associated class probabilities. The experiment showed that YOLO can learns very general representations of objects outperforming other detection methods.

Fuzzy logic was proposed as an inference mechanism to expressing qualitative knowledge and experience with unclear boundaries [17]. The membership function is used to distinguish fuzzy sets. The fuzzy relationships were used to implement rule-based reasoning. It contributes to various application (e.g., the description system whose model is unknown or uncertain and the control object with strong nonlinearity).

However, OpenPose library only estimated the human body pose without posture recognition. Furthermore, posture recognition also suffer from various uncertainty from environment (e.g., viewpoint, illumination and obstacles) and posture (ambiguity of posture and overlap of multiple people). Therefore, in this paper, the information extracted by OpenPose is input into the deep neural network to train the model for recognizing each person's postures. The deep learning model's predictions are evaluated via the metrics (e.g., accuracy, precision, recall).

The recognized postures from the scene may or may be a behavior that require attention (e.g., punch posture can be either a violent "fighting" behavior or just a normal "doing exercise" behavior, squat posture can be either a help needed injured behavior or just a normal "tie shoelaces" behavior). Therefore, the fuzzy logic is proposed in this paper for inferencing/reasoning the behavior category from the recognized posture.

Several practical applications of object recognition and tracking for unmanned vehicle has been proposed from the same research team in this proposed paper. Those include a multi-pose face tracking and recognition system using unmanned aerial vehicle [18], face

recognition and tracking for an indoor security robot [19], detection of temperature, mask wearing, and recognition of human face for a COVID-19 pandemic response robot [20] and object tracking for an unmanned surface vehicle [21]. This article proposes an UAV surveillance system for human body posture recognition and tracking. The proposed system including human tracking, posture recognition and behavior interpretation.

2. Materials and Methods

The scenario of the proposed system is shown in Figure 1. The postures are classified into seven categories. The procedure of the proposed system is illustrated in Figure 2. The image is obtained from the UAV's camera. The proposed system first detects and tracks the people in the image. Then, the people's skeleton is extracted for further postures recognition. Finally, those postures are inferenced for the behavior recognition (normal, violent or need help).

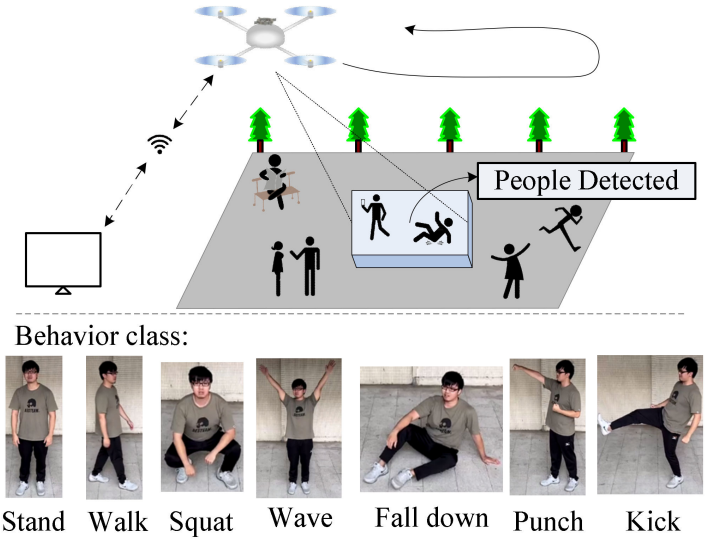

Figure 1. The scenario of the proposed system.

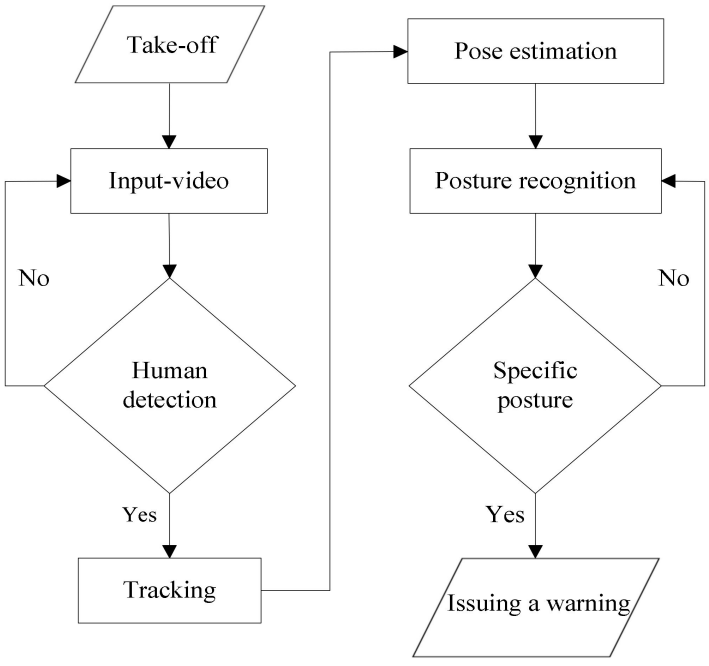

Figure 2. The flowchart of proposed system.

2.1. Human Tracking

Object tracking consists of two parts: object detection and tracking. Simple online and real-time tracking (SORT) [14] uses the Hungarian Algorithm and the Kalman Filter to measure the overlap degree of the frame, which surrounds the object in image space and links data frame by frame. This simple solution shows good results at high frame rates. SORT returns a disproportionately high number of identity transitions, despite its overall good precision and accuracy. Since the used association metric is only reliable when state estimation uncertainty is minimal, therefore SORT tracking suffers from in the presence of occlusions.

Deep SORT is a refinement of the SORT target tracking algorithm. It creates features for subjects using a pre-trained neural network, allowing relations to be made based on feature similarities rather than overlapping. The deep SORT algorithm is extremely powerful and fast, making it the most popular object detection and tracking algorithm [15]. The flow chart of human tracking is shown in Figure 3. Deep SORT processes each frame as follows:

1. The detector gets the bounding box.
2. To generate detection.
3. Kalman quantization prediction.
4. Uses the Hungarian Algorithm to match the predicted track with the detection in the current frame.
5. Kalman Filter update.

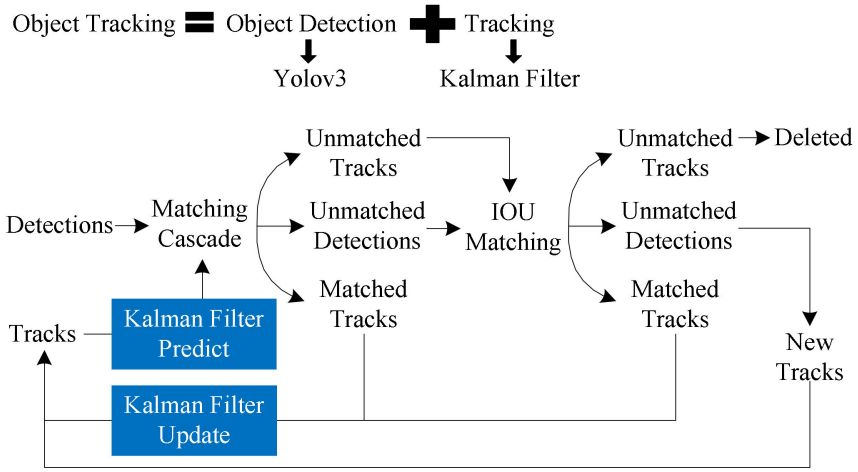

Figure 3. Flow chart of human tracking.

The You Only Look Once (YOLO) framework [16] is adopted for the object detection. The YOLO is one-stage and end-to-end target detection framework with fast speed and good generalization ability. This paper applied YOLOv3, an improved version of YOLOv2 with a similar detection rate from the aspect of accuracy and a three-fold faster. Darknet-53 was used to replace Darknet-19, which used a large number of residual jump layer designs to generate three feature maps of various scales. Furthermore, YOLOv3 performed multi-tag classification using logistic regression, rather than softmax.

YOLO-block, Residual block (resblock-body), 2D convolution layer (conv2D), batch normalization (BN), leaky rectified linear unit (Leaky-ReLU) module and an upsampling layer are the basic modules. The use of the BN layer instead of the dropout, which is used on each conv2D BN Leaky-ReLU (DBL) module, is one of the most noticeable improvements in the YOLOv3 network structure. The YOLOv3 network structure is shown in Figure 4.

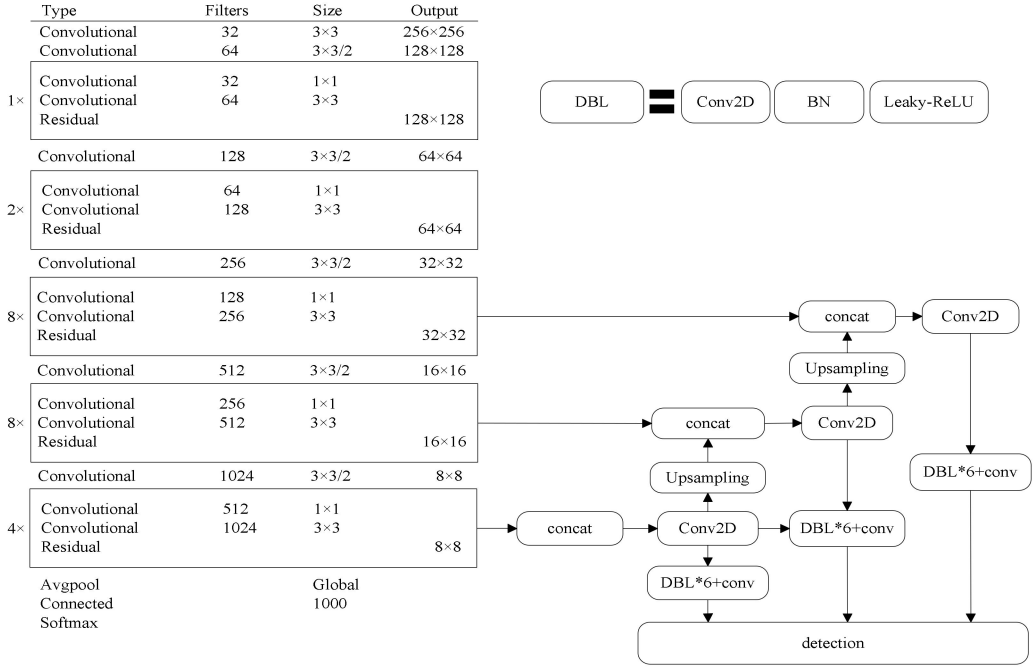

Figure 4. YOLOv3 network structure.

The DBL module, on the other hand, is made up of a conv2D convolution layer, a BN layer and a Leaky-ReLU activation feature, which serves as the foundation for the entire network module. The resblock-body, as the Darknet-53 network's core module, is made up of a zero-padding layer, a DBL module and residual units. The Daknet-53 network is made up of five residual bodies and a DBL module, with each subject in the residual stack having a different coefficient. The feature maps are output to the next residual body after the stacking residuals operation is completed and the feature extraction operation is continued.

Cross-scale prediction, which offers three prediction boxes of varying sizes, is a major advance in YOLOv3. The feature map is generated by the Darknet-53 network's last three residual subjects, which are combined with the feature map in the latter stage through the serrate convolution stack module.

In this way, more features of the detected object can be extracted, reducing the occurrence of detection errors. The YOLOv3 algorithm achieves the advantages of fast detection speed and good generalization ability. Each bounding box is predicted to have four offset coordinates (t_x, t_y, t_w, t_h). The prior value of the bounding box's width and height are linearly connected. The predicted coordinates (b_x, b_y, b_w, b_h) can be obtained as follows:

$$b_x = \sigma(t_x) + c_x, \tag{1}$$

$$b_y = \sigma(t_y) + c_y, \tag{2}$$

$$b_w = p_w e^{t_w}, \tag{3}$$

$$b_h = p_h e^{t_h}, \tag{4}$$

where c_x and c_y are the offset coordinates of the image's upper left corner and p_w and p_h are the bounding box's width and height. These parameters are used to represent the value of the bounding box.

The detector obtains the bounding box, generates the detection result and then uses the Kalman filter to make predictions. The prediction and calculation phases are the two main stages in the Kalman filter. The prediction stage is a method for predicting state variables in a system of interest based on prior knowledge of the state variables and their corresponding prediction noise. It also entails calculating the variance associated with the state variable es-

timation. The measurement level, on the other hand, is a procedure for reading information about state variables at present, while taking measurement noise into account. The next state variable of the system of interest is estimated using a combination of information from the prediction and measurement stages. It also entails calculating a variance correlated with the state variable measurements. The procedure happens recursively.

In the development of an optimization algorithm, a Kalman filter is viewed as a single agent that searches for theoretical global optima solutions. Any agent carries information about the state variable of a system. It represents an individual agent's location in the search space [22]. The equation is shown as follows

$$X_i(t-1) = \left\{ x_i^1(t-1), x_i^2(t-2) \cdots x_i^D(t-1) \right\}, \tag{5}$$

where $x_i^D(t-1)$ is a position of an agent, i is a number of an agent, D is a search space dimension and t are a number of an iteration. In the prediction stage, the equations are shown as follows

$$\hat{X}_i(t) = X_i(t-1), \tag{6}$$

$$\hat{P}(t) = \hat{P}(t-1) + \hat{Q}, \tag{7}$$

where $\hat{X}(t)$ is an agent's expected location. A current variance associated with the prediction, a previous variance associated with the prediction and a prediction noise covariance, which is known as a constant, are $\hat{P}(t)$, $\hat{P}(t-1)$ and $\hat{Q}$, respectively. The search agents are set to move randomly by the use of the predicted location in the measurement point, which is implemented using the equation as follows

$$z_i(t) = \hat{X}_i + A, \tag{8}$$

$$A = \sin(rand \times 2\pi) \times (1 - \exp(|\hat{X}_i(t)|)), \tag{9}$$

where A is used to introduce randomness into the agent's movement. Equation (8) also calculates the quest agent's new location. Equations (10)–(12) are applied, and thus complete the algorithm's cycle.

$$X_i(t) = \hat{X}_i(t) + K \times (z_i(t) - (|\hat{X}_i(t)|)), \tag{10}$$

$$K(t) = \frac{\hat{P}(t)}{\hat{P}(t) + R}, \tag{11}$$

$$P(t) = (1 - K(t)) \times \hat{P}(t), \tag{12}$$

where $X_i(t)$ is the current position of the i agent and K is a Kalman gain, which is defined as in (11). R is a constant that represents the estimation noise covariance. $P(t)$ is the current variance associated with the estimation.

The Kalman filter is applied for motion prediction in multiple objects tracking, and then followed by the Hungarian algorithm to associate each object predicted between frames [23]. The Hungarian Algorithm is a combined optimization algorithm for task allocation. When applied to the tracking problem, it becomes the match between the objects of the previous frame and the next frame. The cost matrix is obtained by combining motion and appearance metrics in deep SORT. The motion metric calculates the difference between predicted Kalman states and newly arrived measurements using Mahalanobis distance. The equation represents the Mahalanobis distance between the j-th detection and the i-th tracker, shown as follows

$$d^{(1)}(i,j) = (d_j - y_i)^T {S_i}^T (d_j - y_i), \tag{13}$$

where d_j is the j-th detection bounding box, and y_i is the position predicted by the Kalman filter for the i-th trajectory. S_i is the variance matrix of the current measurement space of the Kalman filter.

When the uncertainty of the object movement is high, there will be a lot of unmatched Mahalanobis distances, thus making the motion metric Invalid. This is because as the uncertainty of Kalman filter prediction increases, the correlation between states becomes smaller, and the inverse matrix of covariance matrix becomes larger. Therefore, importing the appearance metric is to compare the similarity of the image between the frame predicted by the Kalman filter and the frame obtained by the next frame object detection.

The similarity of the image is to embed the image to 128-dimensions space through the convolutional neural network and limit the length to one, and then calculate the cosine distance. The equation represents the distance between the j-th detection and the i-th tracker, shown as follows

$$d^{(2)}(i,j) = \min\{1 - {r_j}^T r_k^{(i)} | r_k^{(i)} \in R_i\}, \tag{14}$$

$$R_k = \{{r_k}^{(i)}\}_{k=1}^{L_k}, \tag{15}$$

where r_j and ${r_k}^{(i)}$ are the appearance descriptor and the trajectory descriptor. L_k is a constant and R_k is the gallery to store the latest trajectory descriptors. Equations (13) and (14) combine as the complete formula for calculating the correlation metric between the i-th object monitoring and the j-th object detection, shown as follows

$$c_{i,j} = \lambda d^{(1)}(i,j) + (1-\lambda) d^{(2)}(i,j), \tag{16}$$

where λ is the weight. Considering the continuity of motion, the detection result can be filtered and the combination of threshold functions can be defined (17).

$$b_{i,j} = \prod_{m=1}^{2} {b_{i,j}}^{(m)}, \tag{17}$$

where $b_{i,j}$ is the threshold function. Next, the cascade matching is for matching and the algorithm is listed in Algorithm 1.

Algorithm 1: Matching Cascade.

Input: Track indices $T = \{1, \ldots, N\}$, Detection indices $D = \{1, \ldots, M\}$, Maximum age $A_{\max}$
Compute cost matrix $C = [c_{i,j}]$ using Equation (16)
Compute gate matrix $B = [b_{i,j}]$ using Equation (17)
Initialize set of matches $M \leftarrow \emptyset$
Initialize set of unmatched detections $U \leftarrow D$
for $n \in \{1, \ldots, A_{\max}\}$ **do**
    Select tracks by age $T_n \leftarrow \{i \in T \mid a_i = n\}$
    $[x_{i,j}] \leftarrow$ min_cost_matching (C, T_n, U)
    $M \leftarrow M \cap \{(i,j) \mid b_{i,j} \cdot x_{i,j} > 0\}$
    $U \leftarrow U \backslash \{j \mid \sum_i b_{i,j} \cdot x_{i,j} > 0\}$
end for
return M, U

Where T is the object tracking set and D is the object detection set. The C matrix stores the calculation results of the distance between all object tracking and object detection. The B matrix stores all the judgments of whether the object tracking is related to the object detection. After calculation, update M to match successfully set. Remove the successfully matched object detection j from U, and return two sets of M and U. In the final stage, the intersection of the unions is used to match unconfirmed and unmatched trajectories.

2.2. Posture Recognition

OpenPose is a pose estimation algorithm that uses CNN to detect multi-person poses from a single image. It works via the following two branch nets after the extraction feature map by using a VGG19-like network structure: one is a confidence maps network (CMN); and the other is a part affinity fields (PAF) network. The CMN uses a heat map format to predict key points for each body part, while the PAF uses vector maps to reflect the probability of interaction between those key points. To improve prediction accuracy, these two branches are replicated as multiple layers. This system uses OpenPose to detect the skeleton, as shown in Figure 5.

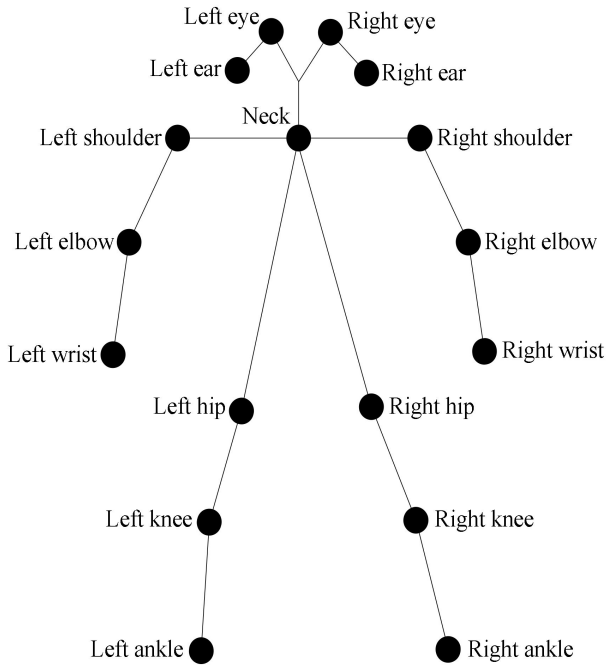

Figure 5. Human skeletal position.

The accuracy of traditional posture recognition is not high, and it cannot solve the uncertainty problem, so this paper uses the method of deep learning. Use OpenPose to obtain the 17 position coordinates of the human skeleton as the input of the CNN. Three models based on CNN for training, namely MobileNet, AlexNet and Visual Geometry Group Network (VGGNet), are applied. These models will be introduced in the following sections.

The first model is MobileNet [24], and its basic unit is the depth separable convolution. Depth separable convolution is actually a factorized convolution, which can be decomposed into two smaller operations: depthwise convolution and pointwise convolution. Depthwise convolution is different from standard convolution. For standard convolution, the convolution kernel is used on all input channels, while depthwise convolution uses different convolution kernels for each input channel. That is to say, one convolution kernel corresponds to one input channel. The pointwise convolution is actually an ordinary convolution, but it uses a 1×1 convolution kernel.

The two operations and standard convolution are shown in Figure 6. For depthwise separable convolution, it first uses depthwise convolution to convolve different input channels separately, and then uses pointwise convolution to combine the above outputs, so that the overall effect is similar to a standard convolution, but will reduce the calculation and model parameter quantity. Here is an analysis of the difference between the depthwise separable convolution and the standard convolution in terms of calculation. The input feature map size is $D_F \times D_F \times M$, and the output feature map size is $D_F \times D_F \times N$, where D_F is the width and height of the feature map, which are consistent and the same. Kernel refers to the number of channels. The standard convolution kernel is $D_K \times D_K$.

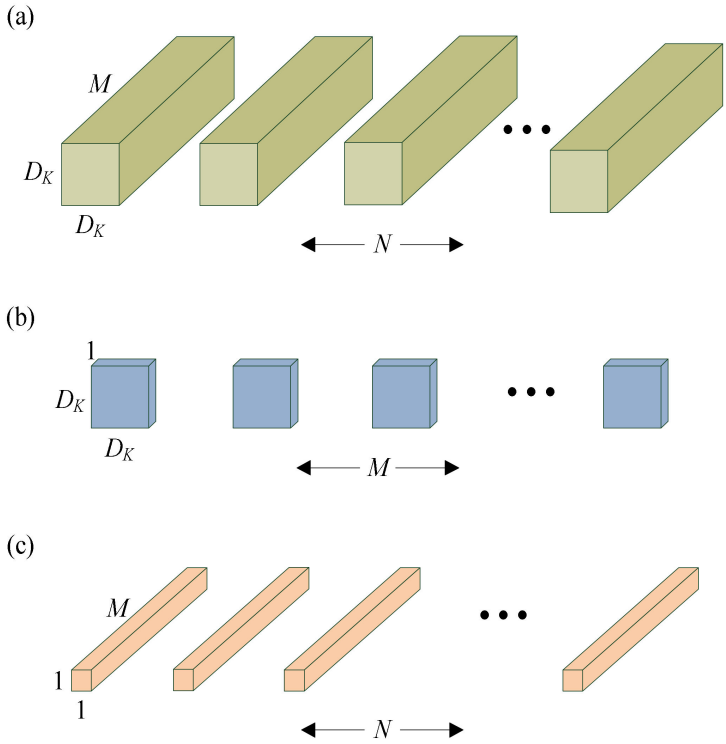

Figure 6. (**a**) Standard convolution filters; (**b**) depthwise convolutional filters; and (**c**) pointwise convolutional filters.

The calculation amount of standard convolution, depthwise convolution and pointwise convolution is as follows

$$C_s = D_K \times D_K \times M \times N \times D_F \times D_F, \tag{18}$$

$$C_d = D_K \times D_K \times M \times D_F \times D_F, \tag{19}$$

$$C_p = M \times N \times D_F \times D_F, \tag{20}$$

where C_s, C_d and C_p represent the calculation amount of standard convolution, depthwise convolution and pointwise convolution, respectively. Therefore, the total calculation amount of depthwise separable convolution is shown as follows

$$C_{ds} = D_K \times D_K \times M \times D_F \times D_F + M \times N \times D_F \times D_F, \tag{21}$$

where C_{ds} is the calculation amount of depthwise separable convolution. Depthwise separable convolution can be compared, and the standard convolution is shown as follows

$$\frac{C_{ds}}{C_s} = \frac{1}{N} + \frac{1}{{D_K}^2}, \tag{22}$$

where the N is relatively large, so if a 3×3 convolution kernel is used, the depthwise separable convolution can reduce the calculation amount by about nine times, compared with the standard convolution. Therefore, MobileNet can be used to reduce the amount of calculation and shorten the training time. The MobileNet structure is shown in Figure 7.

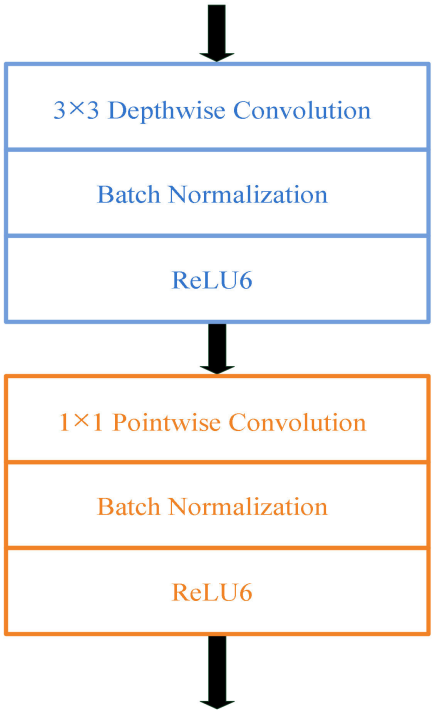

Figure 7. MobileNet structure.

The second model is AlexNet [25], which is an older deep network that was applied to ImageNet and has much higher accuracy than conventional methods. AlexNet demonstrates that success under dynamic models causes CNN to gain popularity. Before AlexNet, most networks used tanh(x) or Sigmoid as the activation feature. The mathematical expression of the two are as follows

$$f(x) = \tanh(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}, \tag{23}$$

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}, \tag{24}$$

where x is the input of the neuron.

However, AlexNet discovered that using the ReLU activation feature in the architecture could speed up the training speed. The mathematical expression of ReLU is as follows

$$f(x) = \begin{cases} x, & \text{if} x \geq 0 \\ 0, & \text{otherwise}, \end{cases} \tag{25}$$

where x is the input of the neuron.

The architecture of AlexNet is shown in Figure 8. There are eight layers in the model. Convolution and max-pooling layers make up the first to fifth layers, while totally linked layers make up the sixth to eighth layers, shown in Figure 9. Dropout is used to minimize model overfitting in AlexNet's first and second layers. The neuron activation value is disabled in the current layer with a certain probability in the dropout layer, which is useful to prevent overfitting. Dropout is also beneficial, as neurons are chosen at random and interdependence between them is minimized, meaning that essential features that are independent of each other are extracted [26].

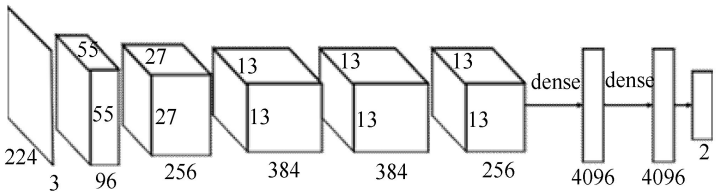

Figure 8. AlexNet architecture.

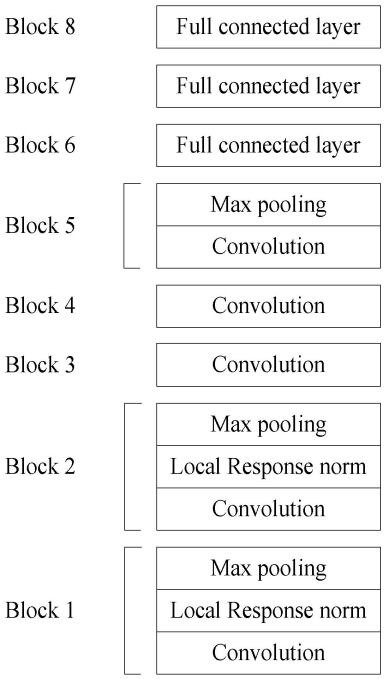

Figure 9. AlexNet model.

To improve the model's generalization ability, AlexNet suggested a local response normalization approach to smooth the output effects of the current layer. Let $a^i{}_{x,y}$ denote that the kernel I source output is applied at position (x, y), and then the ReLU nonlinearity is applied, the output of kernel i at that position $b^i{}_{x,y}$ is given by the expression as follows

$$b^i_{x,y} = \frac{a^i_{x,y}}{\left(k + \alpha \sum_{j=\max(0,\frac{i-n}{2})}^{\min(N-1,\frac{i+n}{2})} \left(a^i_{x,y}\right)^2\right)^{\beta}}, \tag{26}$$

where N is the total number of kernels, n is the size of the normalization neighborhood and α, β, k are hyper-parameters.

The third model is VGGNet [27]. This can be thought of as a more advanced version of AlexNet. A convolutional layer and a completely wired layer make up the whole network. The usage of consecutive 3×3 convolution kernels to replace the larger convolution kernel in AlexNet (11×11, 5×5) is an upgrade of VGG16 over AlexNet. Convolution impacts the receptive field, in addition to the amount of calculation. The former is concerned about whether it is simple to train and process in real time. The latter is linked to parameter updates, function map size, model complexity, parameter number and whether enough features are extracted [28].

Since multiple non-linear layers will increase the depth of the network to ensure a more robust learning mode, and the expense is comparatively small with fewer calculations, using stacked small convolution kernels is better than using large convolution kernels for a given receptive area (the local size of the input picture compared to the output). Furthermore, it is thought that the size of the receptive field obtained by stacking two 3×3

convolutions is equal to a 5×5 convolution, and the size of the receptive field obtained by stacking three 3×3 convolutions is equal to a 7×7 convolution. Under the same stride, the function maps and convolution parameters of different convolution kernel sizes are not significantly different in terms of the amount of calculation, but the larger the convolution kernel, the greater the amount of calculation.

The VGG's key contribution is the use of a tiny convolution kernel (3×3) to build and compare convolutional neural network architectures of different depths. A network of 16–19 layers is sufficient to achieve an acceptable recognition accuracy [29]. VGG16 would finally have j neurons in the last layer to identify j groups. As a result, the total number of input values would be j. Convert these j input values to the likelihood of each group (between 0 and 1), and then use the softmax activation function on the final layer, as shown in the following:

$$y_j = \sigma(Z)_j = \frac{e^{Z_j}}{\sum_{k=1}^{K} e^{Z_k}}, \text{for } j = 1, \ldots, K, \tag{27}$$

where j is the number of categories and Z is the output of the previous layer. The denominator is used as a regular term, and the expression can be obtained as follows:

$$\sum_{j=1}^{K} y_j = 1, \tag{28}$$

where y is the output layer of the neural network. The structure diagram for VGG16/VGG19 shows 16 layers (including 13 convolutional layers and 3 completely connected layers) and 19 layers (including 16 convolutional layers and 3 fully connected layers). The structure diagram is shown in Figure 10.

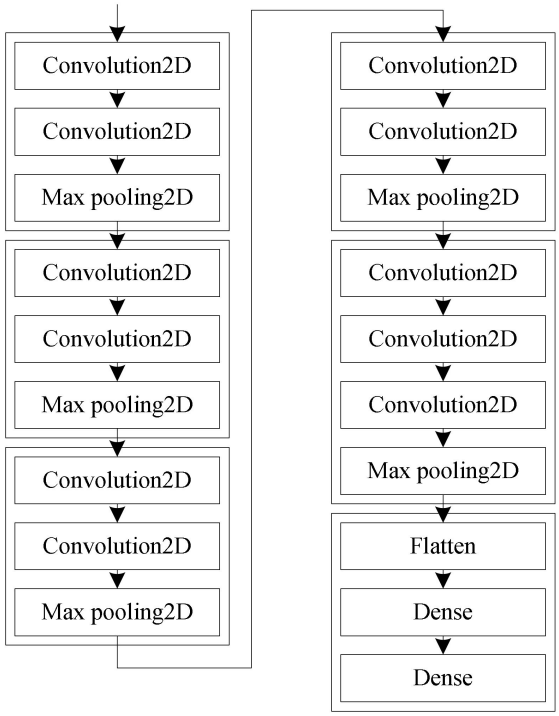

Figure 10. VGG architecture.

2.3. Behaviour Interprtation

The fuzzy logic is applied to infer the poses detected and recognized as shown in Figure 11, whether the pose is normal, needs help or is violent behavior.

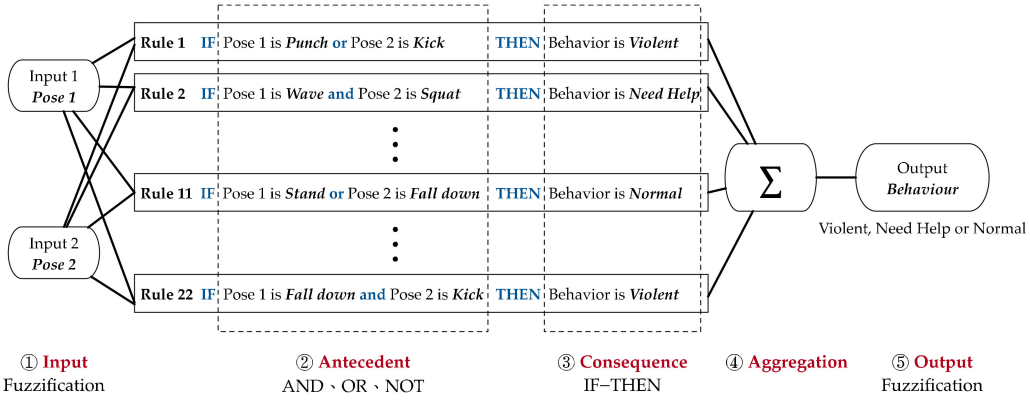

Figure 11. The fuzzy inference processes.

The rules of inference are shown in Table 1. The strength of the antecedent (IF) is obtained by applying the fuzzy operator (AND, OR and NOT). The consequence is further inferred by applying the fuzzy operator IF-THEN (Implication) between the strength of the antecedent and the output membership function.

Table 1. The rules of fuzzy logic.

$Input_1$	IF (Antecedent) Operation	$Input_2$	THEN (Consequence) Output
Punch	or	Kick	Violent
Wave	and	Squat	Help
Wave	and	Fall down	Help
Stand	or	Walk	Normal
Punch	and	Fall down	Help
Punch	and	Squat	Violent
Kick	and	Fall down	Help
Kick	and	Squat	Violent
Stand	or	Wave	Normal
Stand	or	Squat	Normal
Stand	or	Fall down	Normal
Fall down	and	Squat	Help
Squat	or	Stand	Normal
Walk	and	Punch	Normal
Stand	and	Punch	Normal
Fall down	and	Wave	Help
Walk	or	Kick	Normal
Squat	and	Punch	Violent
Squat	and	Walk	Normal
Walk	or	Squat	Normal
Fall down	or	Squat	Help
Fall down	and	Kick	Violent

3. Results

The proposed system is tested in an on-shelf and DIY UAV (refer to the Section 5). Yolov3 framework is applied for the human body detection with a bounding box indicating position of human body and a number represents the number of people detected, and deep SORT algorithm is applied for the continuous tracking.

The skeleton of the human body is extracted by OpenPose for the classification of postures of the human body. The skeleton consists of 17 points on the human body labelled as follows and illustrated in Figure 12.

- 1-right ankle
- 2-right knee
- 3-right hip
- 4-left hip
- 5-left knee
- 6-left ankle
- 7-right wrist
- 8-right elbow
- 9-right shoulder
- 10-left shoulder
- 11-left elbow
- 12-left wrist
- 13-neck
- 14-right ear
- 15-right eye
- 16-left eye
- 17-left ear

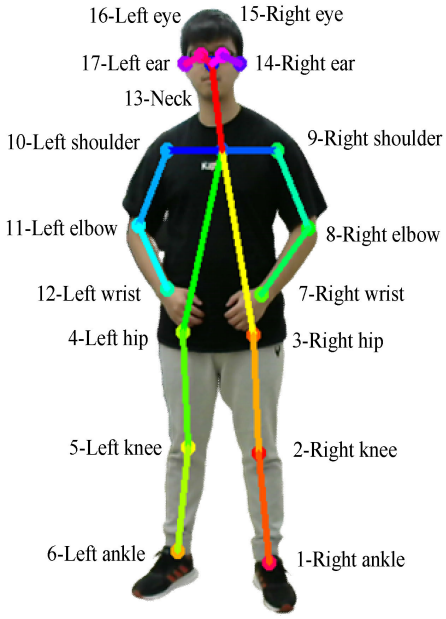

Figure 12. The result of extracting the human skeletal position.

The postures are classified into seven categories as follows and the posture label number indicated the degree of danger (larger the number, more dangerous). The YOLOv3 object detection framework is adopted to detect the people in the image, and the Deep sort algorithm is applied to track the human after detection.

- 7-kick
- 6-punch
- 5-fall down
- 4-wave
- 3-squat
- 2-walk
- 1-stand

The training data collected for each posture are listed in Table 2. Each data is an image contains 17 points as in Figure 12, and each point is composed of position in x-axis and y-axis. Therefore total 35 values for each data (17 point's position (x, y) plus the category

label) fed as the inputs to the neural network model. The output of the neural network model is the seven classes of postures.

Table 2. Number of data used for training and testing.

Class	No. of Training Data	No. of Testing Data
Stand	591	253
Walk	543	379
Squat	714	194
Fall down	590	278
Wave	619	273
Punch	690	305
Kick	835	438

The metrics to evaluate model's predictions are applied using the testing data. Those common metrics include accuracy, precision, recall, F_1 score. The precision shows the ratio of correct prediction of positive over all prediction as positive. The recall shows the ratio of correct prediction of positive over actual is positive. These evaluation metrics are obtained from a confusion matrix, as shown in Figure 13. The optimal value for each of those metrics is listed in Table 3.

Actual \ Prediction	Positive	Negative
Positive	**True Positive (TP)** correct	**False Negative (FN)** incorrect
Negative	**False Positive (FP)** incorrect	**True Negative (TN)** correct

Figure 13. Confusion matrix.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{29}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{30}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{31}$$

$$\text{F}_1 = \left(\frac{\text{Precision}^{-1} + \text{Recall}^{-1}}{2}\right)^{-1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{32}$$

Table 3. The optimal values for each evaluation metric of the model.

Class	Precision	Recall	F_1-Score
Stand	0.725	0.846	0.781
Walk	0.804	0.858	0.83
Squat	0.611	0.825	0.702
Fall down	0.908	0.654	0.719
Wave	0.697	0.868	0.773
Punch	0.661	0.889	0.758
Kick	0.999	0.511	0.676

The results from the training and testing stage show the model is fitted (good generalization capability, neither underfitting nor over fitting) as shown in Figure 14. The confusion matrix of the model is shown in Figure 15.

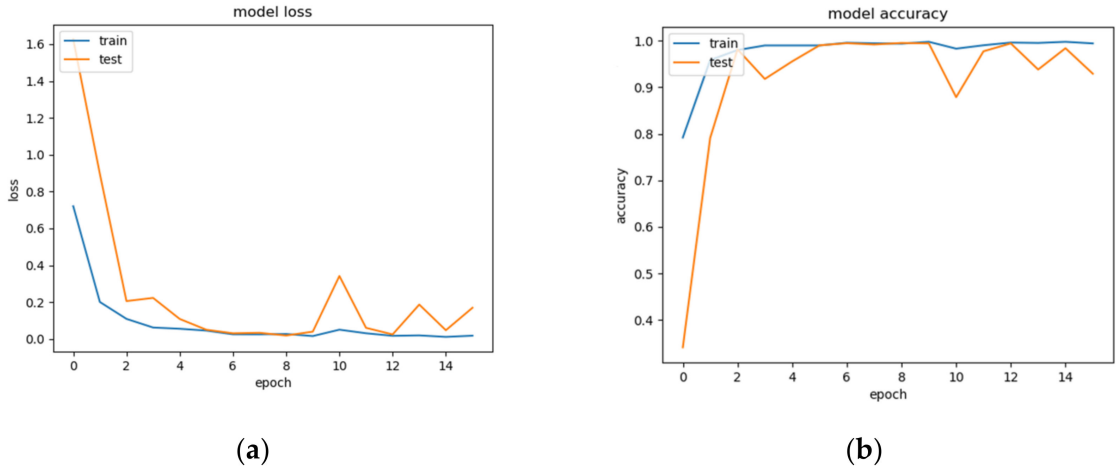

Figure 14. Performance of model during training and testing stage as: (**a**) Loss; (**b**) Accuracy.

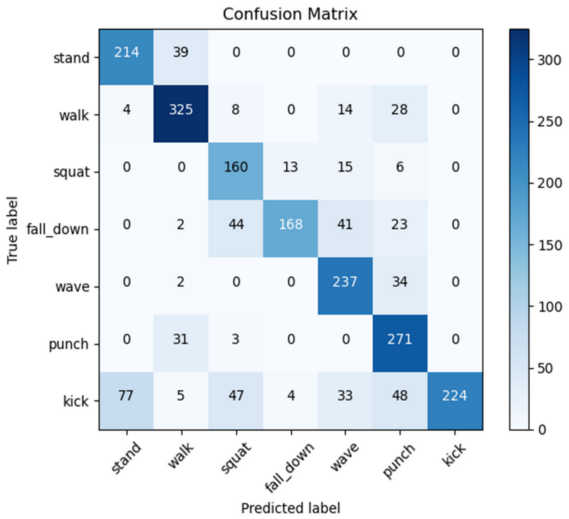

Figure 15. The confusion matrix of the model.

The resultant P-R (Precision-Recall) curve is generated by moving the classification threshold from high to low as shown in Figure 16. Using only precision and recall corresponding to a certain point (classification threshold) cannot fully measure effectiveness of model. Only through overall performance of P-R curve, the model can be more comprehensively evaluated

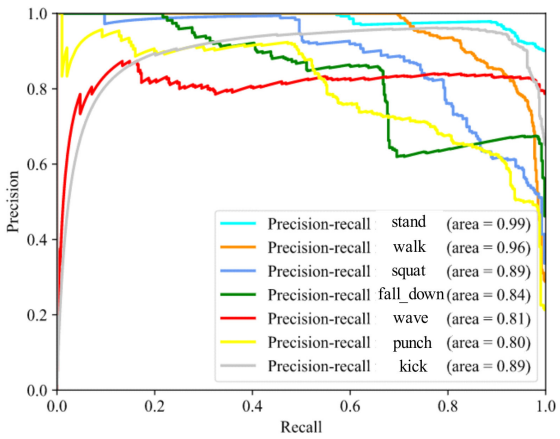

Figure 16. P-R curve of the model.

The ROC (Receiver Operating Characteristic) curve plots TPR (True Positive Rate) vs. FPR (False Positive Rate) under different classification thresholds as shown in Figure 17. The AUC (Area Under ROC Curve) provides an aggregate measure of performance across all possible classification thresholds as listed in Table 4. Larger the AUC, classifier is more likely to rank real positive samples in the front to achieve better classification performance.

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}, \tag{33}$$

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{34}$$

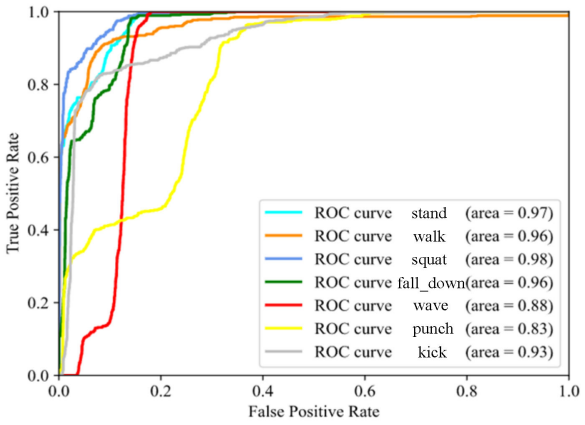

Figure 17. ROC curve of the model.

Table 4. Optimal AUC of the model.

Class	AUC
Stand	0.97
Walk	0.96
Squat	0.98
Fall down	0.96
Wave	0.88
Punch	0.83
Kick	0.93

The model is trained offline and deployed to the UAV processor for testing. If the testing result is not optimal, then the testing data is fed into the training set for offline training again. The result of posture recognition (seven postures) is shown in Figure 18.

Fuzzy logic process is further applied to interpret those recognized multiple postures for corresponding behavior (situation) as in Figure 19 (indoor). Three classes of behaviors as normal in Figure 19a, violent in Figure 19b,c and help needed are shown in in Figure 19d.

Outdoor behavior interpretation is shown as in Figure 20. Three kinds of behaviors as Normal in Figure 20a, Violent in Figure 20b,c and Need help in Figure 20d.

The results of outdoor posture recognition and tracking upon occlusion are shown as in Figure 21. The results show the system can continuously recognize and track when there is partial/full occlusion

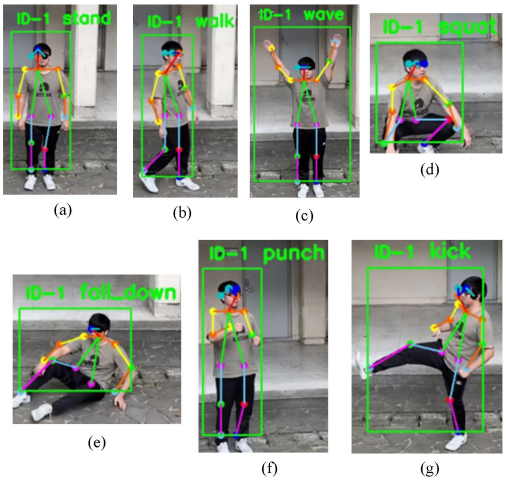

Figure 18. The result of posture recognition. (**a**) stand; (**b**) walk; (**c**) wave; (**d**) squat; (**e**) fall down; (**f**) punch; and (**g**) kick.

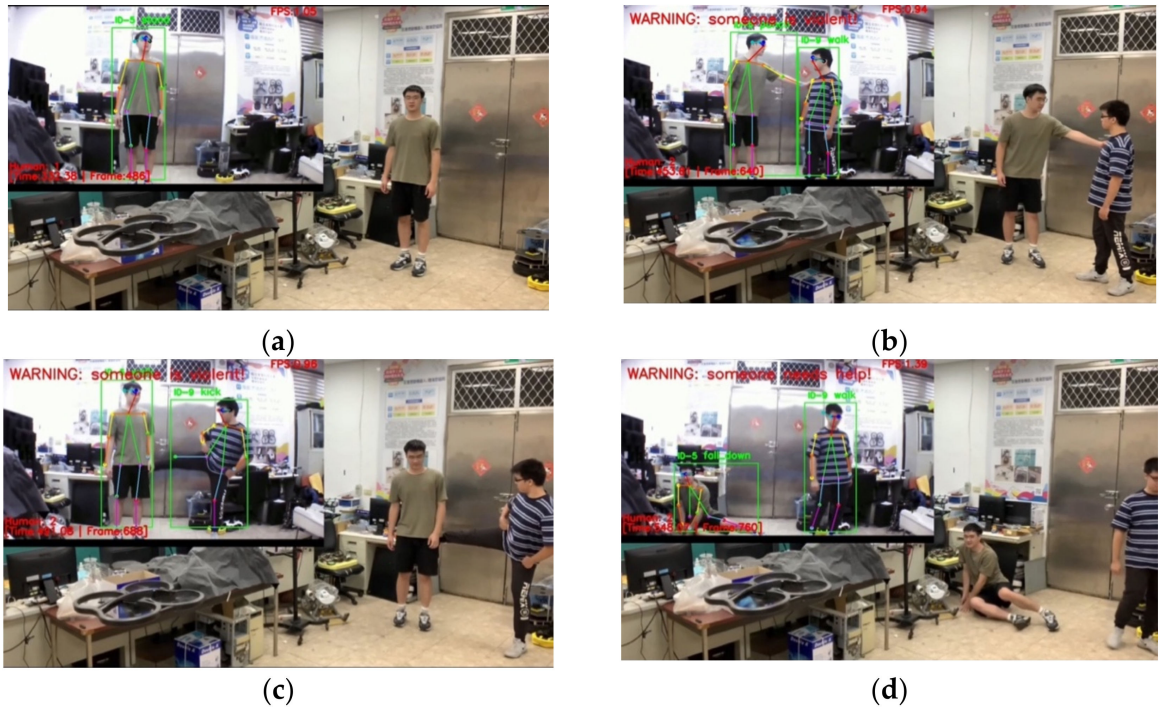

Figure 19. Indoor Behavior interpretation from the multiple postures recognized as: (**a**) Normal (stand posture); (**b**) Violent (one is stand and another is punch posture); (**c**) Violent (one is stand and another is kick posture); (**d**) Help needed (one is walk pose and another is fallen down posture).

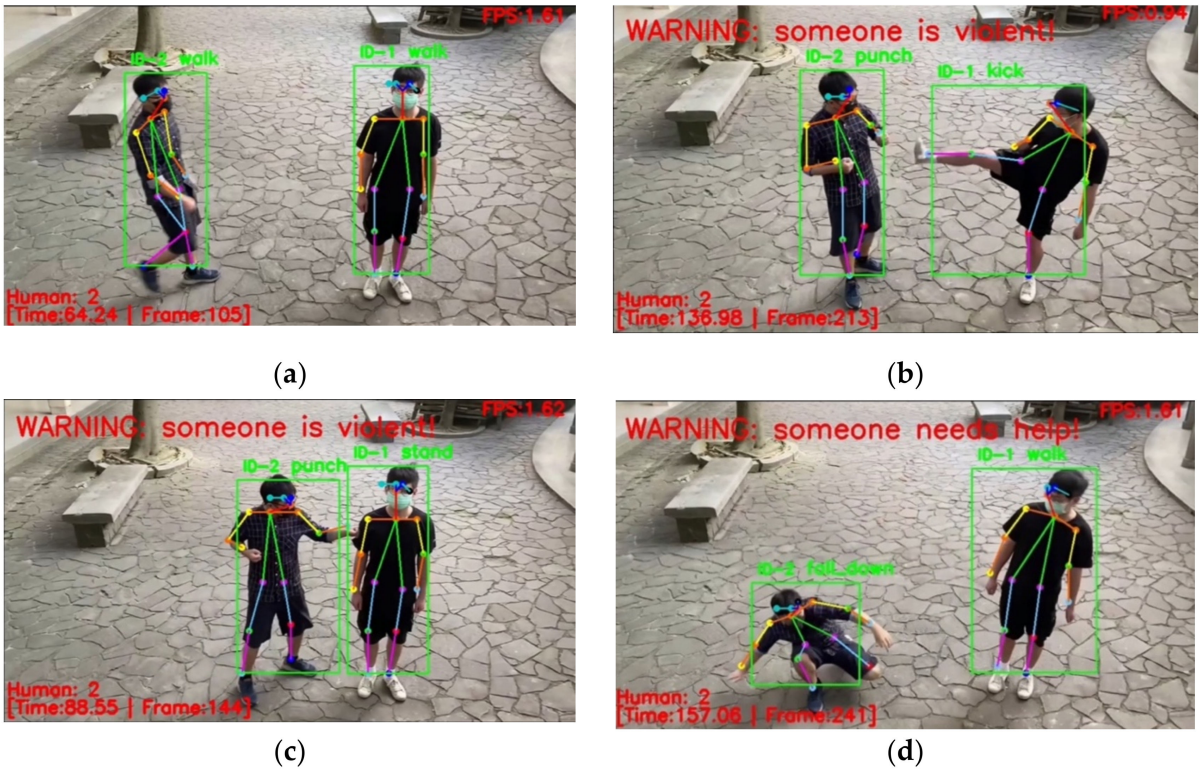

Figure 20. Outdoor behavior interpretation from the multiple postures recognized as: (**a**) Normal (both persons are walk posture); (**b**) Violent (one is punch and another is kick posture); (**c**) Violent (one is stand and another is punch posture); (**d**) Help needed (one is walk pose and another is fallen down posture).

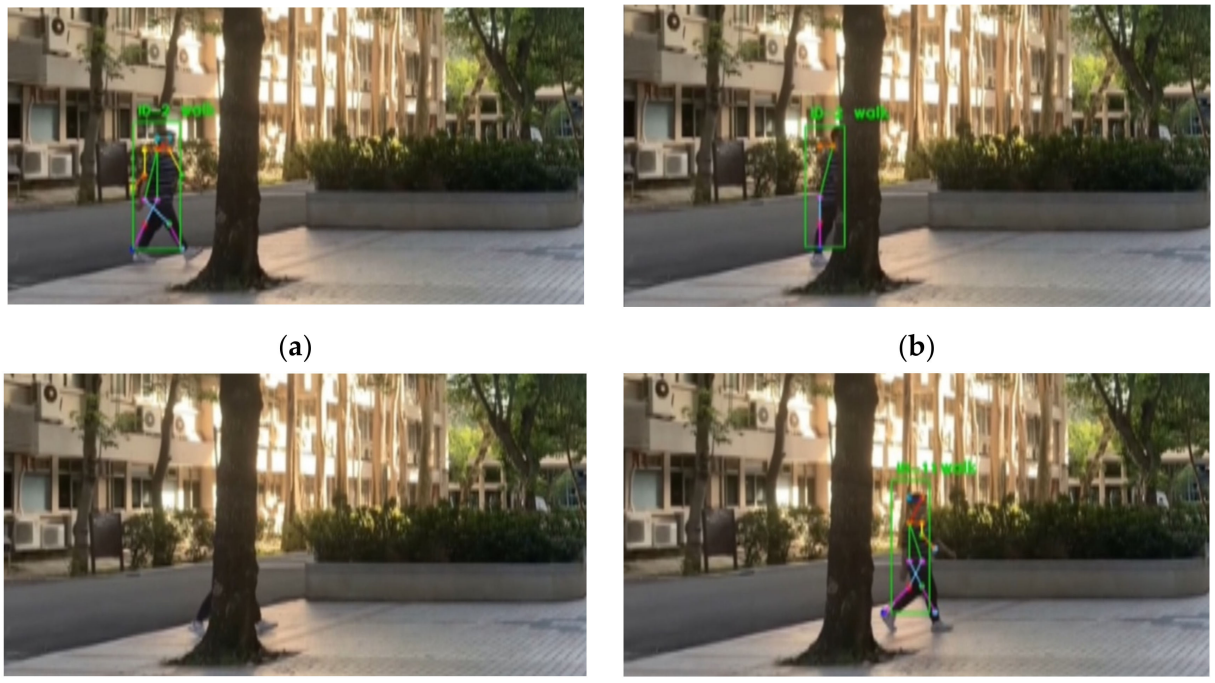

Figure 21. Outdoor posture recognition and tracking upon occlusion as: (**a**) Walk before occlusion; (**b**) Partial occlusion; (**c**) Full occlusion; (**d**) Walk out of occlusion.

4. Discussion Conclusions

UAVs' situational awareness is important for many applications. In these applications, one of the critical tasks are to locate and recognize a specific target and respond promptly. This paper proposes a human posture recognition/tracking and behavior interpretation system for UAV, which can be used for search, rescue or surveillance tasks.

The skeleton of the human body is extracted by OpenPose and used to for human posture detection (right ankle, right knee, right hip, left hip, left knee, left ankle, right wrist, right elbow, right shoulder, left shoulder, left elbow, left wrist, neck, right ear, right eye, left eye and left ear).

The Yolov3 is adopted for human detection (prior to human posture detection) and human posture recognition (kick, punch, fall down, wave, squat, walk and stand). The deep SORT is adopted for human tracking. The fuzzy logic is applied for the behavior interpretation (normal, violet and help needed) from the recognized postures.

The accuracy of posture recognition achieved results with an accuracy of 95.2% while maintaining real-time performance. However, the posture is actually a dynamic movement consisting of sequential static poses, so classification on by single frame is not a good solution. In the future, the consideration of using recurrent neural network models to classify postures with dynamic sequential joint data can further improve the performance.

As an important aspect of the deep learning model for human detection, the model in this paper does not rely on the assumption of the orientation of the bounding box or the aspect ratio. The deep learning model used for posture recognition does not rely on the assumption of the orientation of the bounding box or the aspect ratio which outperform than the traditional approach.

Overall, this paper demonstrated that the system onboard the UAV is a viable and effective solution for situation-aware tasks which involve the detection/recognition/tracking of human posture and interpret the behavior. The proposed onboard system of UAV is compact and efficient which avoid the requirement of a robust network system for remotely processing.

5. Patents

The patents resulting from the work reported in this manuscript are as follows: "Torque Generating Device and Multirotor Aerial Vehicle", Patent No. TW I749799, Taiwan, December 2021.

Author Contributions: Conceptualization, M.-F.R.L.; methodology, M.-F.R.L., Y.-C.C. and C.-Y.T.; software, C.-Y.T.; validation, M.-F.R.L., Y.-C.C. and C.-Y.T.; formal analysis, M.-F.R.L.; investigation, M.-F.R.L.; resources, M.-F.R.L.; data curation, Y.-C.C. and C.-Y.T.; writing—original draft preparation, Y.-C.C. and C.-Y.T.; writing—review and editing, M.-F.R.L.; visualization, Y.-C.C. and C.-Y.T.; supervision, M.-F.R.L.; project administration, M.-F.R.L.; and funding acquisition, M.-F.R.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Ministry of Science and Technology (MOST) in Taiwan grant number [108-2221-E-011-142-] and the Center for Cyber-Physical System Innovation from the Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Herrero, M.J.; Perez-Fortes, A.P.; Escavy, J.I.; Insua-Arevalo, J.M.; Horra, R.D.; Lopez-Acevedo, F.; Trigos, L. 3D Model Generated from UAV Photogrammetry and Semi-automated Rock Mass Characterization. *Comput. Geosci.* **2022**, *163*, 105121. [CrossRef]
2. Sun, Y.; Ma, O. Automating Aircraft Scanning for Inspection or 3D Model Creation with a UAV and Optimal Path Planning. *Drones* **2022**, *6*, 87. [CrossRef]
3. Skondras, A.; Karachaliou, E.; Tavantzis, I.; Tokas, N.; Valari, E.; Skalidi, I.; Bouvet, G.A.; Stylianidis, E. UAV Mapping and 3D Modeling as a Tool for Promotion and Management of the Urban Space. *Drones* **2022**, *6*, 115. [CrossRef]
4. Yoo, M.; Na, Y.; Song, H.; Kim, G.; Yun, J.; Kim, S.; Moon, C.; Jo, K. Motion Estimation and Hand Gesture Recognition-Based Human–UAV Interaction Approach in Real Time. *Sensors* **2022**, *22*, 2513. [CrossRef] [PubMed]
5. Saini, N.; Bonetto, E.; Price, E.; Aamir, A.; Black, M.J. AirPose: Multi-View Fusion Network for Aerial 3D Human Pose and Shape Estimation. *IEEE Robot. Autom. Lett.* **2022**, *7*, 2. [CrossRef]
6. Psiroukis, V.; Espejo-Garcia, B.; Chitos, A.; Dedousis, A.; Karantzalos, K.; Fountas, S. Assessment of Different Object Detectors for the Maturity Level Classification of Broccoli Crops Using UAV Imagery. *Remote Sens.* **2022**, *14*, 731. [CrossRef]
7. Abeywickrama, D.B.; Cirstea, C.; Ramchurn, S.D. Model Checking Human-Agent Collectives for Responsible AI. In Proceedings of the IEEE International Conference on Robot and Human Interactive Communication, New Delhi, India, 14 October 2019; pp. 1–8.
8. Lina, W.; Ding, J. Behavior Detection Method of OpenPose Combined with Yolo Network. In Proceedings of the International Conference on Communications, Information System and Computer Engineering, Kuala Lumpur, Malaysia, 3–5 July 2020; pp. 326–330.
9. Cao, Z.; Hidalgo, G.; Simon, T.; Wei, S.-E.; Sheikh, Y. OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 172–186. [CrossRef] [PubMed]
10. Qiao, S.; Wang, Y.; Li, J. Real-time human gesture grading based on OpenPose. In Proceedings of the International Congress on Image and Signal Processing, BioMedical Engineering and Informatics, Shanghai, China, 14–16 October 2017; pp. 1–6.
11. D'Antonio, E.; Taborri, J.; Palermo, E.; Rossi, S.; Patanè, F. A markerless system for gait analysis based on OpenPose library. In Proceedings of the IEEE International Instrumentation and Measurement Technology Conference, Dubrovnik, Croatia, 25–28 May 2020; pp. 1–6.
12. Hirasawa, Y.; Gotoda, N.; Kanda, R.; Hirata, K.; Akagi, R. Promotion System for Home-Based Squat Training Using OpenPose. In Proceedings of the IEEE International Conference on Teaching, Assessment, and Learning for Engineering, Takamatsu, Japan, 8–11 December 2020; pp. 984–986.
13. Huang, C.; Nguyen, M.H. Robust 3D Skeleton Tracking based on OpenPose and a Probabilistic Tracking Framework. In Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Bari, Italy, 9 October 2019; pp. 4107–4112.
14. Bewley, A.; Ge, Z.; Ott, L.; Ramos, F.; Upcroft, B. Simple online and realtime tracking. In Proceedings of the IEEE International Conference on Image Processing, Phoenix, AZ, USA, 25–28 September 2016; pp. 3464–3468.
15. Wojke, N.; Bewley, A.; Paulus, D. Simple online and realtime tracking with a deep association metric. In Proceedings of the IEEE International Conference on Image Processing, Beijing, China, 17–20 September 2017; pp. 3645–3649.
16. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In *Proceedings of the 29th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, 26 June–1 July 2016*; IEEE Computer Society: Las Vegas, NV, USA, 2016.
17. Zadeh, L.A. *Fuzzy Logic*; Springer: New York, NY, USA, 2012; pp. 1177–1200.
18. Lee, M.-F.R.; Li, Y.-C.; Chien, M.-Y. Real-time face tracking and recognition using the mobile robots. *Adv. Robot.* **2015**, *29*, 187–208. [CrossRef]
19. Lee, M.-F.R.; Chen, Y.-C.C. COVID-19 Pandemic Response Robot. *Machines* **2022**, *10*, 351. [CrossRef]
20. Lee, M.-F.R.; Shih, Z.-S. Autonomous Surveillance for an Indoor Security Robot. *Processes* **2022**, *10*, 2175. [CrossRef]
21. Lee, M.-F.R.; Lin, C.-Y. Object Tracking for an Autonomous Unmanned Surface Vehicle. *Machines* **2022**, *10*, 378. [CrossRef]
22. Azwan, A.; Razak, A.; Jusof, M.F.M.; Nasir, A.N.K.; Ahmad, M.A. A multiobjective simulated Kalman filter optimization algorithm. In Proceedings of the IEEE International Conference on Applied System Invention, Chiba, Japan, 13–17 April 2018; pp. 23–26.
23. Sahbani, B.; Adiprawita, W. Kalman Filter and Iterative-Hungarian Algorithm Implementation for Low Complexity Point Tracking as Part of Fast Multiple Object Tracking System. In Proceedings of the 2016 6th International Conference on System Engineering and Technology (ICSET), Bandung, Indonesia, 3–4 October 2016.
24. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
25. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
26. Titoriya, A.; Sachdeva, S. Breast Cancer Histopathology Image Classification using AlexNet. In Proceedings of the International Conference on Information Systems and Computer Networks, Mathura, India, 21–22 November 2019; pp. 708–712.
27. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.

28. Saiharsha, B.; Abel Lesle, A.; Diwakar, B.; Karthika, R.; Ganesan, M. Evaluating Performance of Deep Learning Architectures for Image Classification. In Proceedings of the International Conference on Communication and Electronics Systems, Coimbatore, India, 10–12 June 2020; pp. 917–922.
29. Cheng, G.; Ma, C.; Zhou, P.; Yao, X.; Han, J. Scene classification of high-resolution remote sensing images using convolutional neural networks. In Proceedings of the International Geoscience and Remote Sensing Symposium, Beijing, China, 10–15 July 2016; pp. 767–770.