Advanced Human Activity Recognition Using Pose Estimation and Deep Learning Techniques

Haider Rasheed Hassan
Computer department
University of Diyala
Diyala, Iraq
scicompms222307@uodiyala.edu.iq

Dheyab Salman Ibrahim
Computer department
University of Diyala
Diyala, Iraq

Ziyad Tariq Mustafa Al-Ta'i
Computer department
University of Diyala
Diyala, Iraq

***Abstract*—Human activity recognition (HAR) is a prominent research topic in computer vision. This topic has extensive applications in developing applications for human-machine interactions, monitoring, and various other fields. With the notable progress in research, several methods have been proposed to distinguish distinct types of human movements by utilizing color, depth, inertia, and skeletal data. To enhance the accuracy of human activity recognition and overcome the constraint of existing deep learning approaches that mainly rely on video frame inputs, we provide The solution being suggested using MediaPipe, a platform that provides pre-trained machine-learning models specifically designed for human pose estimation. These models will precisely identify and monitor essential body joints and movements while doing activities. Subsequently, the system will examine the identified poses to compute significant angles and distances for feature extraction. Following that, deep learning techniques such as Artificial Neural Networks (ANN) and 1D-convolutional Neural Networks (1D-CNN) are employed to categorize human behaviour. The proposed system's performance was assessed using our dataset to detect and categorize different human activities. The ANN and 1D-CNN models showed superior performance in recognizing human activity, with the ANN achieving an accuracy of 99.41 % and the 1D-CNN achieving an accuracy of 99.58 %.**

Keywords—HAR, Deep Learning, 1D-CNN, ANN, Human Pose Estimation, MediaPipe

I. INTRODUCTION

As the interaction between humans and computers grows, it has become increasingly crucial for computers to detect human behavior and actions. This recognition of human activity or behavior is essential for many applications, including caring for older adults, fitness, home automation, psychological analytical research, detecting and preventing violent acts, security, and many more [1].

Human activity recognition (HAR) extraction from video data has gained significant interest, primarily because of the abundance of large video datasets and the progress made in deep learning methodologies. Video-based human activity recognition (HAR) offers distinct advantages compared to image datasets and sensor modalities such as accelerometers and gyroscopes. This is because video-based HAR can provide a more comprehensive perspective of human activities, encompassing temporal and spatial data. Video footage can correctly capture the context and environmental information that aid in detecting human behaviors [2].

In recent years, thorough studies have been conducted into recognizing human behavior using deep learning approaches, which have yielded a solution. Deep learning methods produce feature maps using artificial neural networks [3]. Deep neural networks have significantly advanced computer vision, natural language processing, and robotics. Nevertheless, there is still a need to enhance deep learning algorithms for superior performance because Human motion is intricate, and its analysis is influenced by factors such as chaotic backgrounds, diverse lighting conditions, unstable image acquisition, and insufficient pattern classes [4].

This paper introduces a novel model that utilizes human pose estimation techniques using video-based data. Specifically, it employs MediaPipe posture, an ML solution that accurately tracks body poses and extracts (33) 3D-landmarks from RGB video frames [5]. As a result, we have developed a rapid and efficient detector that demonstrates strong performance in extracting features.

We aim to determine the most effective deep learning algorithm for human action detection. We will use two algorithms, 1D-CNN and ANN, which are well-suited for training, testing, and accurately classifying actions. The main contributions to the suggested model include the following:

- Built new dataset encompassing various activities conducted in both indoor and outdoor environments.
- Developing new model for high speed and low processing requirements has been created to recognize human activity from videos.

The organization of this paper is as follows: Section 2 contains the relevant research, Section 3 specifies our strategy, and Section 4 contains the experiments and analysis. In conclusion, Section 5 summarizes the entirety of the article.

II. LITERATURE REVIEW

As mentioned in the introduction, one of the most extensively studied areas in the field of computer vision is HAR. Various methodologies have been presented based on using DL algorithms in the last years to solve the HAR problem [6].

In 2022, Basly et al. [7], The authors proposed a deep temporal residual system for recognizing daily living activities. They utilized a deep residual convolutional neural network (RCN) to preserve distinctive visual features related to appearance and a long short-term memory (LSTM) neural network to capture the extended temporal progression of actions. The model was applied to two well-known datasets, MSRDailyActivity3D and CAD-60, for human activity recognition. The suggested approach attains an accuracy of 91.65 % on the MSRDailyActivity3D dataset and 91.18 % on the CAD-60 dataset. A limitation of this study is that the proposed system's performance was only evaluated on two benchmark datasets. These datasets were captured in an indoor environment, specifically a living room, and they only represent a subset of possible scenarios in the real world.

In 2023, Mathew et al. [8] Two deep learning approaches, single-frame CNNs and convolutional (LSTM), were used to recognize human activities in the movie. Both models were trained and evaluated using UCF50, a standardized action recognition dataset.

The dataset contains videos of diverse human activities. However, the UCF50 dataset trained only three activities: 'PullUps', 'WalkingWithDog', and 'PlayingGuitar'.This was done due to resource and time restrictions and slow model training, as the UCF50 dataset is enormous. They scored 99.8% on the UCF50 dataset. Their performance was also assessed using their own data set of "Jumping", "Walking", and "Sitting". They were 83.33 % accurate.

In 2023, Mohan et al. [2], in this study, hybrid models, such as Convolutional LSTM (ConvLSTM) and Long-term Recurrent Convolutional Networks (LRCN), have been used to enhance the accuracy of Human Activity Recognition (HAR) on video datasets. The evaluation of the models is conducted using standard video datasets, specifically the UCF50 dataset, which consists of 50 distinct action types. However, they used only a subset of actions due to limitations in computing power and storage capacity. The activity was identified using the ConvLSTM and LRCN models, with accuracies of 83.46 % and 92.13 %, respectively.

In 2023, U. Dedhia et al. [9], this research uses Google Mediapipe to create a virtual fitness trainer by tracking user movement with posture estimate techniques. It tracks user motions by recognizing exercise-specific bodily markers. Angles and landmarks are used to calculate the user's performance and feed it into a machine-learning model like Logistic Regression, Support Vector Machine, naive Bayes, Decision Tree, and Artificial Neural Network. The dataset utilized was obtained from Kaggle. Although data was available for three exercises, the emphasis was mainly on bicep curls. Furthermore, user-supplied data is utilized to ensure comprehensive training and assessment of bicep curls. A constraint of this study is the limited size of the dataset. The maximum accuracy achieved in the Logistic Regression (tune) model is 0.99.

III. The Proposed Model

The proposed model has main four entities: Data Collection, Feature Extraction, Pre-processing, and Classification employing Deep Learning techniques, illustrated in Fig. 1.

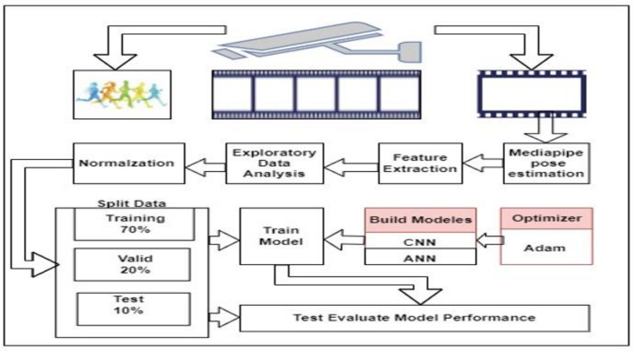


Fig. 1. The System Model

A. Dataset Collection Stage

The authors constructed the dataset locally specifically for this study. This dataset comprises real-time video recordings of individuals engaged in various activities in front of the camera. The dataset consists of 10 individuals who are high school students between the ages of 18 and 25. The filming was done with a Sony HDR_CX405 camera. The camera is situated at a height of 130cm above the ground and a distance of 450 cm from the subject. A total of 120 video clips were utilized in this research, with each person contributing twelve clips. Table (1) provides information about these data sets.

TABLE I. Provides information about these data sets

Table 1: Dataset Description		
No.	Dataset Details	Description
1	Number of person	10
2	The number of videos per person	12
3	Number of classes (Activities)	12
3	Frame Rate	25 frame per second
4	Camera type	Sony HDR_CX405
5	File format	MP4
6	video display time	About 3 minute

The dataset also contains many diverse lighting situations in a diverse environment. The data set includes 12 activities, and each activity took place in an environment (indoor and outdoor). These activities are: (raising legs, running, sitting, squatting, standing, walking, clapping, jogging, boxing, hand waving, kicking, and shooting), and Fig. 2 shows Sample activities within the data set.

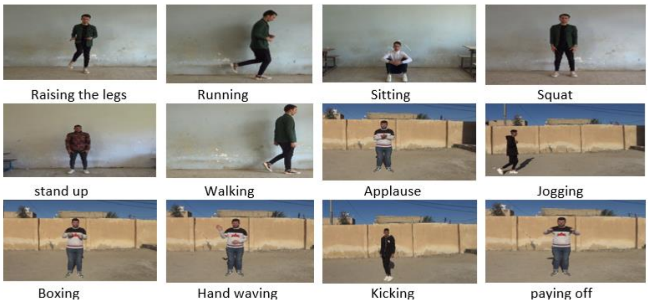
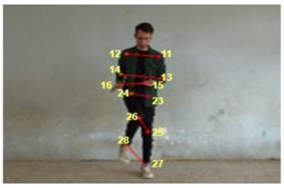

Raising the legs
Running
Sitting
Squat
stand up
Walking
Applause
Jogging
Boxing
Hand waving
Kicking
paying off

Fig. 2. Samples of activities within the dataset

B. Feature Extraction Stage

During the feature extraction stage, the Media Pipe model is utilized to identify the 2D key joint positions for each front and side video frame. This article used MediaPipe Pose (MPP), a cross-platform framework that Google developed to obtain 2D coordinates of human joints in each image frame.MediaPipe Pose constructs pipelines and analyzes cognitive information in video format using machine learning (ML). MPP utilizes a BlazePose to extract 33 2D landmarks on the human body, BlazePose is an efficient machine-learning framework that achieves real-time execution on mobile phones and PCs [10]. To estimate poses and activities, our study used 12 specific landmarks, with indices 11, 12, 13, 14, 15, 16, 23, 24, 25, 26, 27, and 28. These landmarks are depicted in Fig. 3.



No.	Name of key joint positions		
11	Left Shoulder.	12	Right Shoulder.
13	Left Elbow.	14	Right Elbow.
15	Left Wrist.	16	Right Wrist.
23	Left Hip.	24	Right Hip.
25	Left Knee.	26	Right Knee.
27	Left Ankle.	28	Right Ankle.

Fig. 3. Names of the 12 key joint position

The position estimation component of the proposed system predicts the locations of all major joint positions (12) for each individual. This gives the video a thick box and places the person performing the activity toward the detector in the first frame of the video. The tracker then assumes control and categorizes the landmark's points inside the designated bounding box (ROI: Region of Interest). The ROI from the previous frame is used by the tracker to continue running on any more video frames. Our work involves using a hypothetical point, referred to as the mid(x,y), positioned both above and below the individual in space. This unique approach allows us to identify and analyze various features, which in this case pertain to angles and distances. These features constantly change as the person engages in different activities, with their body parts moving in different directions over time. The hypothetical point is calculated by calculating the midpoint (up) between the middle of the shoulders and (down) the middle of the hip, as shown in the following algorithm (1).

```
Algorithm 1: calculate a hypothetical point (up and down)
Input: Frame
Output: hypothetical point (up and down)
Step1: Resize the Frame
Step2: get height, width from Frame
Step3: Utilize a pose landmark model to obtain left-side landmarks
          [left-Shoulder, left-Hip]
Step4: Utilize a pose landmark model to obtain right-side landmarks
          [right-Shoulder, right-Hip]
Step5: Find midpoint up point // Get coordinates for each landmark
Step5-1: Get coordinates x=(left-Shoulder.x   + right-Shoulder.x) / 2
            Get coordinates Y=(right-Shoulder.x + right-Shoulder.x) / 2
Step5-2 midpoint= (coordinates x, coordinates y)
Step6: up point =(0, coordinates y)
Step7: Find midpoint up point // Get coordinates for each landmark
Step7-1: Get coordinates x=(left- Hip.x   + right- Hip.x) / 2
            Get coordinates Y=(right- Hip.x + right- Hip.x) / 2
Step7-2 midpoint= (coordinates x, coordinates y)
Step8: up down=(height, coordinates y)
Step9: return (up point, up down)
```

We calculate all the angles that form the main joints with the hypothetical point, as shown in Fig. 4 and Algorithm 2.

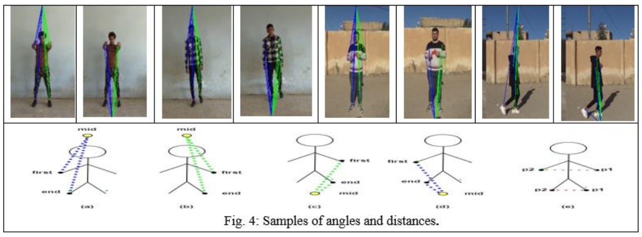


Fig. 4. Samples of angles and distances.

The following algorithm (3) explains the extraction of features from video frames.

```
Algorithm 2:  Calculate Angle
Input : Three points in 2D space ( first, mid, end)
Output: angle
Step1: Calculate Numerator
         numerator =mid.y*(first.x-end.x)+first.y*(end.x-mid.x)+end.y*(mid.x – first.x)
Step2: Calculate denominator
         denominator = (mid.y-first.x)*( first.x -end.x) +(mid.y – first.y)*(first.y – end.y)
Step3: Calculate Angle
Step3-1: find the angle in radians
            $ang = arctan\ (\frac{numerator}{denominator})$
Step3-2: Convert the angle from radians to degrees
            $ang\ =\ ang\ *\ \frac{180}{\pi}$
Step4: If the resulting angle is negative
          if ang < 0 then
             ang = 180 + ang
Step5: return ang
```

```
Algorithm (3) Extraction of the  “Features”
Input: file video
Output: Features (angles and distances).
Step1: Open the video file reading
Step2:  Loop Over Frames
Step2-1: Resize the Frame
Step2-2: landmark detection in MediaPipe
Step2-2-1: Utilize a pose landmark model to obtain the left-side landmarks
              [ left -Shoulder, left-Elbow, left-Wrist, left-Hip, left-Knee, left-Ankle].
Step2-2-2: Utilize a pose landmark model to obtain the right-side landmarks
              [ right-Shoulder, right-Elbow, right-Wrist, right-Hip, right-Knee, right-Ankle]
Step2-3: Define a hypothetical point mid.up(x1 ,y1) up the person and mid.down( x2,y2) down
the person
Step2-4: Extraction features of distances left-side and right-side
            For each landmarks-left, landmarks-Right in left-side, right-side do
                // Get coordinates for each landmarks
                P1= [landmarks-left.x, landmarks-right.y ] ,
                P2= [landmarks-left.x, landmarks-right.y ],
                Distance= ((p2[0] – p1[0]) *2 + (p2[1] –p1[1]) *2) * 0.5
                Features. Add(Distance)
            End for
Step2-5: Extraction features of angles left-side with mid.up and mid.down
Step2-5-1: Feature angles hypothetical point mid.up
              For each landmarks-left, landmarks-Right in left-side, right-side do
                  // Get coordinates for each landmarks
                  first= [landmarks-left .x,landmarks-left.y],end=[landmarks-left.x,landmarks-left.y ]
                  angle-left-up= call calculate angle1 (first,mid.up,end)
                  first=[landmarks-Right.x,landmarks-Right.y],end=[landmarks-Right.x,landmarks-Right.y ]
                  angle-Right-up= call calculate angle1 (first,mid.up,end)
                  Features.Add(angle-left -up)
                  Features.Add(angle-Right-up)
              End for
Step2-5-2: Feature angles hypothetical point mid.down
              For each landmarks-left, landmarks-Right in left-side, right-side do
                  // Get coordinates for each landmarks
                  first= [landmarks-left .x,landmarks-left.y],end=[landmarks-left.x,landmarks-left.y ]
                  angle-left- down = call calculate angle1 (first,mid. down,end)
                  first=[landmarks-Right.x,landmarks-Right.y],end=[landmarks-Right.x,landmarks-Right.y ]
                  angle-Right- down = call calculate angle1 (first,mid. down,end)
                  Features.Add(angle-left - down)
                  Features.Add(angle-Right- down)
              End for
Step2.3: end Over Frames
Step3: return Features
```

C. Pre-processing Stage

This stage's inputs consist of features retrieved from the 2D key join points. The pre-processing stage comprises two sub-steps: data cleansing through Exploratory Data Analysis (EDA) and Data normalization.

- Cleaning Data

This stage involves assessing and removing outliers from the input dataset to enhance data comprehension and optimize classification model performance. Exploratory data analysis is a method used to search for outliers in large amounts of data. The EDA employs algorithm (4) to address outliers in the data [11].

```
Algorithm (4): Elminate the Outlier
Input: Dataset
Output: New Dataset
Begin
Step 1: For each column in a dataset, Do
Step 2: Determine the lower and upper range from the column.
Step 2-1: Arrange in ascending order (column)
Step 2-2: Determine the quantile for each column
            Q1= Column. Quantile (0.25)
            Q3= Column. Quantile (0.75)
```

```
Step 2-3: Calculate lowerand upper range
          IQR = Q3-Q1
          Lower range = Q1-(1.5*IQR)
          Upper range = Q3+(1.5*IQR)
Step 3: Modify the data in the column with the updated range values.
Step 3-1: For each value in the data:
          If value  <  lower range then
              New value ← lower range
              New dataset ←  Update the value in the column
          If value  >  upper range then
              New value ← upper range
              New dataset ←  Update the value in the column
Step 4:        End for
Step 5:   End for
Step 6: Return New Dataset
End
```

- Data Normalization

It involves converting the data into a format that falls into a specific range, such as [0.0, 1.0], to generalize data values. The goal of normalizing the data is to assign equal weight to each feature. Normalization is often used, particularly when the input variables have varied scales, to guarantee the accuracy of the predictive modeling and help hasten the learning process. Min-Max Normalization was used in the data normalization process. This method involves applying linear transformations to the primary data. Assume that max(x) and min(x) reflect the highest and lowest values for a set of features, where the value is given to the feature (x), (z^') is the normalized value, and has the formula as seen in (1) below [12].

$$z' = \frac{x - min(x)}{max(x) - min(x)} \quad (1)$$

D. Classification based on DL Stage

Before commencing this stage, the dataset comprises around 51000 samples as shown in the Fig. 5. Subsequently, it is partitioned into 70 % for training, 20 % for validation, and 10 % for testing. We construct the human activity recognition system using two models, specifically ANN and 1D-CNN.

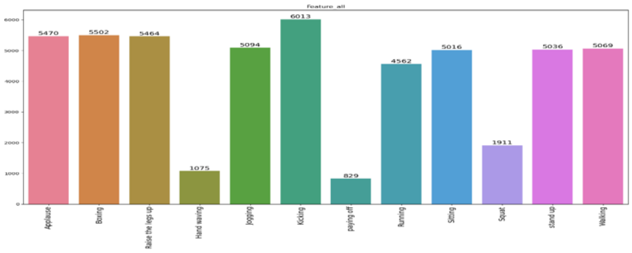


Fig. 5. Distribution of features across each class

- ANN Structure

(ANNs) are computational models that mimic the information-processing capabilities of the human brain. ANNs are trained (or learned) by experience with suitable learning exemplars, not by programming.

The ANN architecture used in the model comprises of three layers: an input layer, an output layer, and a hidden layer, as depicted in Fig. 6. The input layer consists of 128 neurons, while the hidden layer is composed of four layers of 225, 1042, 10, and 16 neurons, respectively. The Artificial Neural Network (ANN) applies a non-linear adjustment to the input of the Rectified Linear Unit (ReLU). It is utilized after the input and concealed layers. It computes using the (2).

$$\boldsymbol{eLU}(\boldsymbol{X}) = \boldsymbol{f}(\boldsymbol{x}) = \max(0.\boldsymbol{x}) =$$

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{cases} \boldsymbol{x_i} & \boldsymbol{x_i} \geq 0 \\ 0 & \boldsymbol{x_i} < 0 \end{cases} \quad (2)$$

The output layer consists of 12 neurons, which corresponds to the number of classes. The softmax` activation function is utilized for multi-class classification, following equation (3).

$$\boldsymbol{f}(\boldsymbol{x_i}) = \frac{\boldsymbol{e^{x_i}}}{\sum_{\boldsymbol{j=1}}^{\boldsymbol{k}} \boldsymbol{e^{x_j}}} \quad (3)$$

where e^{x_i} standard exponential function for input vector, k number of class, and e^{x_j} standard exponential function for output vector

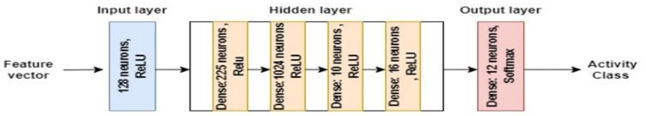


Fig. 6. ANN architecture

- 1D-CNN Structure

Deep learning technique CNN has been widely utilized for computer vision tasks including image categorization and object detection. CNN has neuron layers, pooling layers, and fully linked layers. Convolutional layers scan images or videos with learnable filters. The pooling layers decrease the size and keep just important elements. Fully connected layers classify data using extracted features. We employ 1D-CNN, which are more advantageous and preferable compared to 2D-CNNs due to their lower complexity.

The 1D-CNN architecture used in the model comprises multiple layers, as depicted in Fig. 7. The initial convolutional layer of the model utilizes a collection of 512 filters to process the input features. Each filter has dimensions of 1x1. The Rectified Linear Unit (ReLU) activation function is employed after each level of convolutional layers and with fully connected layers. After each level of convolutional layers, a max pooling layer is applied to the previous layer's output. The pool size utilized for max pooling is (2,2). The model's second convolutional layer utilizes a collection of 256 filters, each with dimensions of 1x1.

Next, a dropout layer was implemented to mitigate overfitting by randomly deactivating 25 % of the neurons. A batch normalization layer is inserted into the model following the dropout layer. This layer stabilizes the learning process by normalizing the output of the preceding layer. The two-dimensional matrix input is subsequently transformed into a vector through a layer known as Flatten. Following this, two fully connected (dense) layers are incorporated into the model. The initial completely connected layer consists of 2048 units. The second layer consists of 1024 units. The output layer serves as the ultimate layer in the neural network. The number of nodes in this layer corresponds to the number of classes in the classification problem, namely the number of human activities in our model. The activation function employed for this layer is the softmax function, which transforms the layer's output into a probability distribution across the classes.

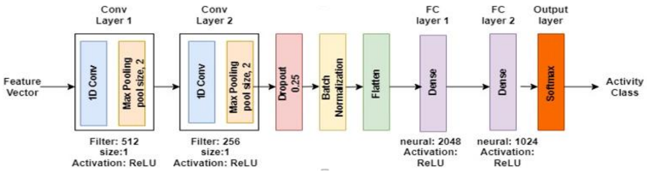


Fig. 7. 1D- CNN architecture

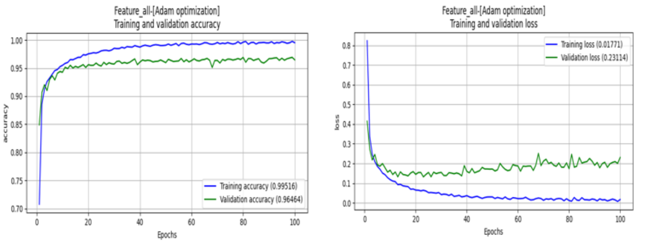
In ANN and 1D-CNN. A machine is designed to acquire knowledge from a dataset and is anticipated to enhance its performance gradually. When the model receives input, a function is applied to it, and it undergoes a series of transformations through several layers, resulting in an output value. The model compares the generated output with the actual output and calculates the difference. To minimize the discrepancy, backpropagation propagates the output back into the model. The model iteratively updates the weights and continues this process until convergence. This drives us to search for an algorithm to expedite learning and produce optimal results. Optimization algorithms are fundamental for enabling a machine to learn from its experiences. It computes gradients and seeks to minimize the loss function. Several optimization strategies implement learning. The algorithm utilized in the current study is (Adam) the term "adaptive moments" is the source of its derivation. It is a hybrid of the rmsprop and momentum optimization algorithms. The update operation exclusively considers the differentiable form of the gradient and incorporates a bias correction method [13] provides a detailed analysis of the Adam algorithm. The loss function evaluates the network's performance in accomplishing its designated goal. Cross-entropy loss, often known as log loss, quantifies the effectiveness of a classifier, And its equation is shown below:

$$Cross\ Entopy = -\sum_{i}^{c} a_i \log(p_i)$$

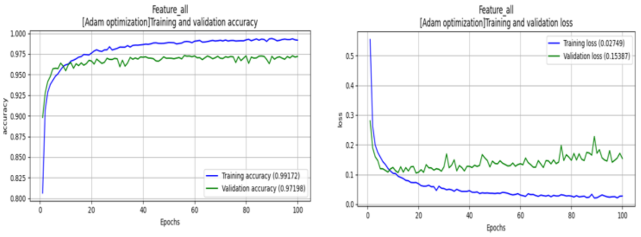
where c represents the number of classes, a_i represents the actual value, and p_i represents the anticipated value.

IV. Experimental Results and Disscusion

In this section, we present the recognition results of the proposed system using two models. (ANN and 1D-CNN) for classifying 12 classes using the collected features dataset with indoor and outdoor environment. Fig. 8 exhibits the accuracy and loss of the implemented models on the training and validation per epoch (epochs = 100).



(a) ANN



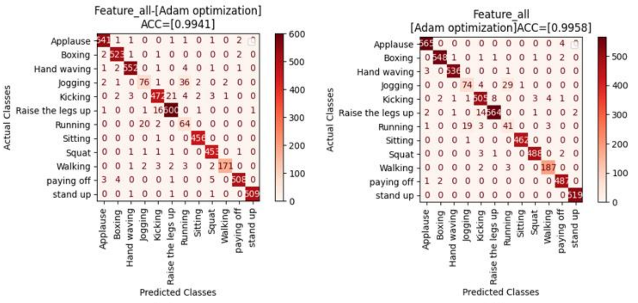
(b) CNN

Fig. 8. The measure of accuracy and loss as a function of epochs for the training set (blue) and validation set (green)

In the first column of the previous figure, we observe that the accuracy values for the implemented models remain stable over the epochs. The training accuracy for the ANN model is approximately 0.99516, and the validation accuracy is around 0.96464. For the 1D-CNN model, the training accuracy is about 0.99172, and the validation accuracy is around 0.97198. These results were obtained after 100 epochs. The second column shows that the training loss value reaches an approximate value of 0.01771, while the testing loss is around 0.23114 in the ANN. The training loss value converges to roughly 0.02749, whereas the validation loss is around 0.15387 in a 1D-CNN model trained for 100 epochs.

The 1D-CNN model has slightly superior validation accuracy (97.20 %) compared to the ANN model (96.46 %). Nevertheless, the disparity is negligible, and both models attain high accuracy. The 1D-CNN model has a validation loss of 0.15387, which is lower than the validation loss of the ANN model, which is 0.23114.

Fig. 9 shows confusion matrix-based ANN and 1D-CNN classification system performance. Confusion matrices assess classifier predictions on a dataset. Diagonal units reflect true positives as the classifier evaluates, while off-diagonal items are mislabeled. Thus, higher confusion matrix diagonal values improve accuracy. The CNN model's 0.9958 % accuracy exceeds ANN's 0.9941 %.



(a)

(b)

Fig. 9. The confusion matrixes for Recognition System based on; (a) ANN (b) 1D-CNN

Fig. 9 shows a comparison between the performance of the ANN and CNN models. This comparison is based on the results of the proposed model, which are shown in Table (2). This analysis aims to determine which model produced the highest classification results.

TABLE II. Performance of the Classification models

Class	ANN				1D-CNN			
	Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Applause	0.99687	0.98364	0.98723	0.98543	0.99785	0.98776	0.99297	0.99036
Boxing	0.99687	0.98124	0.98866	0.98493	0.99804	0.99275	0.98917	0.99096
Hand waving	0.99647	0.98571	0.98221	0.98396	0.99863	0.99443	0.99259	0.99351
Jogging	0.98668	0.76	0.63333	0.69091	0.98923	0.77895	0.68519	0.72906
Kicking	0.98786	0.94779	0.92913	0.93837	0.99079	0.94925	0.9619	0.95553
Raise legs up	0.99079	0.95694	0.96774	0.96231	0.99412	0.98258	0.96575	0.97409

Class	ANN				1D-CNN			
	Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Running	0.98629	0.57143	0.74419	0.64646	0.98844	0.55405	0.61194	0.58156
Sitting	0.99882	0.98915	0.99781	0.99346	0.99961	0.99784	0.99784	0.99784
Squat	0.99647	0.9721	0.98908	0.98052	0.99785	0.98986	0.98785	0.98886
Walking	0.99687	0.98276	0.92935	0.95531	0.99765	0.96392	0.97396	0.96891
paying off	0.99628	0.98833	0.97505	0.98164	0.99765	0.98185	0.99388	0.98783
stand up	0.99922	0.99804	0.99414	0.99609	0.99961	0.99616	1	0.99808
Average	0.994124	0.926428	0.926493	0.924949	**0.995789**	**0.930783**	**0.92942**	**0.929716**

According to Table 5, both models performed well in the activity recognition process, although the CNN model outperformed the other model. The "Accuracy value" is 0.995789, the "precision" is 0.930783, the "Recall" is 0.92942, and the “F1 Score” is 0.929716. The ANN model achieved an accuracy value of 0.994124, a precision of 0.926428, a recall of 0.926493, and an F1 Score of 0.92494949.

Several investigations have focused on the recognition of human activity, employing various methodologies and strategies that have been utilized in previous years. The suggested system demonstrated superior accuracy compared to earlier studies, as demonstrated in Table (3).

TABLE III. COMPARISON OF THE PROPOSED SYSTEM AND RELEVANT APPROACHES

Ref	Classification algorithm	Dataset	Accuracy value
Basly et al. (2022)	RCN and LSTM	MSRDailyActivity3D CAD-60	91.65% 91.18%
Mathew et al. (2023)	CNN and LSTM	UCF50 Own dataset	99.8% 83.33%
Mohan et al. (2023)	ConvLSTM LRCN	UCF50	83.46% 92.13%
U.Dedhia et al. (2023)	LR, SVM, Naïve Bayes, Decision Tree, and ANN	Dataset from Kaggle User data	99%
Our Proposed System	ANN 1D-CNN	New Dataset	99.41% 99.58%

The table above clearly demonstrates that our system performed better than previous systems, except for Mathew et al. who achieved a 99.8 % accuracy using UCF50. However, it is important to note that they only used three out of the 50 activities due to resource and time constraints. When they used their system with private data, which also included only three activities ("jumping," "walking," and "sitting"), they achieved an accuracy of 83.33 %. In comparison, our system utilized our own dataset with twelve activities in both indoor and outdoor environments, achieving accuracies of 99.41 % and 99.58 % in ANN and CNN, respectively. Also, we observed that Didia et al. reached a high level of accuracy, close to 99 %, in their system. However, their study specifically focused on bicep curls, and one weakness of their research is the small dataset.

V. CONCLUSION

HAR utilizing ANN and 1D-CNN models and MediaPipe framework present a promising solution for predicting activity in a video the model has not seen before. The ANN and 1D-CNN conventional networks have effectively addressed many computer vision challenges. ANN and 1D-CNN are highly efficient in carrying out the classification process. MediaPipe Pose (MPP) is a versatile framework created by Google for extracting the 2D coordinates of human joints in every video frame. This amalgamation of methodologies seeks to augment the precision and resilience of video categorization.

In this study, MPP is suitable for our use case since it can automatically extract features from video frames by calculating angles and distances between key points; the two models discussed were trained and tested on features extracted from the video dataset we created for this experimentation. While both models performed ideal in recognition tests, the 1D-CNN model proved more accurate. We aim to expand our newly created dataset to incorporate more videos featuring individuals of diverse ethnic backgrounds in our future endeavors. Because the suggested models can be more effective when applied to a more extensive dataset, we would also like to see them expanded for a larger dataset like Kinetics 700. In addition, we are interested in investigating the potential applications of these models in a surveillance system for financial institutions and airports, where they may be used to record and identify the actions of lone persons. The authorities in charge of these areas can be notified through an alarm system if they observe risky behaviors like "jogging," "falling," or "stealing."

REFERENCES

[1] Z. Yu and W. Q. Yan, “Human Action Recognition Using Deep Learning Methods,” Int. Conf. Image Vis. Comput. New Zeal., vol. 2020-Novem, pp. 2–7, 2020, doi: 10.1109/IVCNZ51579.2020.9290594.

[2] G. K. Mohan, “Recognizing Human Activity Using Hybrid Models of CNN and LSTM in Deep Learning,” Int. J. Food Nutr. Sci., vol. 11, no. 12, pp. 1663–1674, 2023, doi: 10.48047/ijfans/v11/i12/178.

[3] A. Çalışkan, “Detecting human activity types from 3D posture data using deep learning models,” Biomed. Signal Process. Control, vol. 81, no. March, pp. 1–7, 2023, doi: 10.1016/j.bspc.2022.104479.

[4] A. C. Cob-Parro, C. Losada-Gutiérrez, M. Marrón-Romera, A. Gardel-Vicente, and I. Bravo-Muñoz, “A new framework for deep learning video based Human Action Recognition on the edge,” Expert Syst. Appl., vol. 238, no. October 2023, 2024, doi: 10.1016/j.eswa.2023.122220.

[5] T. Lynn, “Pose Estimation Algorithms: History and Evolution,” pp. 1–14, 2023, [Online]. Available: https://blog.roboflow.com/pose-estimation-algorithms-history/

[6] H. C. Nguyen, T. H. Nguyen, R. Scherer, and V. H. Le, “Deep Learning for Human Activity Recognition on 3D Human Skeleton: Survey and Comparative Study,” Sensors, vol. 23, no. 11, pp. 1–33, 2023, doi: 10.3390/s23115121.

[7] H. Basly, W. Ouarda, F. E. Sayadi, B. Ouni, and A. M. Alimi, “DTR-HAR: deep temporal residual representation for human activity recognition,” Vis. Comput., vol. 38, no. 3, pp. 993–1013, 2022, doi: 10.1007/s00371-021-02064-y.

[8] S. Mathew, A. Subramanian, and S. Pooja, “Human Activity Recognition Using Deep Learning Approaches: Single Frame Cnn and Convolutional Lstm”.

[9] U. Dedhia, P. Bhoir, P. Ranka, and P. Kanani, “Pose Estimation and Virtual Gym Assistant Using MediaPipe and Machine Learning,” 2023 Int. Conf. Network, Multimed. Inf. Technol. NMITCON 2023, no. February, 2023, doi: 10.1109/NMITCON58196.2023.10275938.

[10] U. Dedhia, P. Bhoir, P. Ranka, and P. Kanani, “Pose Estimation and Virtual Gym Assistant Using MediaPipe and Machine Learning,” 2023 Int. Conf. Network, Multimed. Inf. Technol. NMITCON 2023, no. September 2023, 2023, doi: 10.1109/NMITCON58196.2023.10275938.

[11] C. H. Yu, “Exploratory data analysis in the context of data mining and resampling.,” Int. J. Psychol. Res., vol. 3, no. 1, pp. 9–22, 2010.

[12] C. Fan, M. Chen, X. Wang, J. Wang, and B. Huang, “A Review on Data Preprocessing Techniques Toward Efficient and Reliable Knowledge Discovery From Building Operational Data,” Front. Energy Res., vol. 9, no. March, pp. 1–17, 2021, doi: 10.3389/fenrg.2021.652801.

[13] X. Jiang, B. Hu, S. Chandra Satapathy, S.-H. Wang, and Y.-D. Zhang, “Fingerspelling Identification for Chinese Sign Language via AlexNet-Based Transfer Learning and Adam Optimizer,” Sci. Program., vol. 2020, no. 1, p. 3291426, 2020.