International Journal on Robotics, Automation and Sciences

Exploring Activities of Daily Living Among the Elderly through Machine Learning Techniques

Josiah Wey Tsen Lim, Tee Connie* and Michael Kah Ong Goh

***Abstract* – Activities of daily living (ADLs) is a term that is used to describe the activities performed in everyday life that involves the motion of the human body such as eating, walking, and sitting. ADLs can be used to determine the state of elderly people as a decline in ADL performance will generally mean a decline in the human body. It can act as an early indicator if an elderly person is experiencing underlying illness or health issue. This project aims to detect five different ADLs which are eating, cooking, sweeping, walking, and sitting and standing. A dataset was collected from twenty individuals performing each ADL at two different angles, a front view and a side view. A computer vision-based human pose estimation technique is used to extract the human body keypoints. These keypoint values are then processed and fit into multiple deep learning models for analysis. In this study, five different deep learning models namely LSTM, Bi-LSTM, CNN, RNN and Transformer models have been evaluated. The performance of each model is analysed and discussed. It was determined that the CNN model performed the best achieving a categorical accuracy of 82.86%.**

Keywords*— *Elderly Safety, Activities of Daily Living, Machine Learning, Motion, Computer Vision.

I. INTRODUCTION

Activities of daily living (ADLs) is generally used to describe the movement and motion used in everyday life such as eating, sweeping and walking. One of the main issues faced by the aging population nowadays is social isolation. Once their children grow up and leave their homes, the parents are usually left to live by themselves without anyone else caring for them. When an elderly individual is left by themselves, small health issues such as chronic pain and vision loss may go unnoticed. Without treatment, these health issues can lead to accidents such as the elderly individual falling down the stairs. This is where the monitoring of ADLs is very important as it can be used to assess the medical and health aspects of an elderly individual. A change in the way an ADL is performed can be an early indicator for an underlying health issue.

This paper collects recordings of twenty elderly individuals performing five different ADL tasks at a front and side angle. These ADLs are walking, eating, standing and sitting, cooking and sweeping. The keypoint values of the body parts of the subject for each action are extracted and collected. The collected actions are kept to 30 frames per action to easier streamline the process. The collected dataset is then used in five different machine learning models and the results are analysed and discussed. A few interesting findings can be obtained once the research was concluded namely, how the machine learning models interact with the dataset. It was determined that the amount of movement or unique motion of the arms and legs contributed heavily to how well ADLs are differentiated from others while actions that only

*Corresponding Author email: tee.connie@mmu.edu.my ORCID: 0000-0002-0901-3831
Josiah Wey Tsen Lim is with Faculty of Information Science and Technology, Multimedia University, Melaka, Malaysia (e-mail: 1191103308@student.mmu.edu.my).
Tee Connie is with Faculty of Information Science and Technology, Multimedia University, Melaka, Malaysia (e-mail: tee.connie@mmu.edu.my).
Michael Kah Ong Goh is with Faculty of Information Science and Technology, Multimedia University, Melaka, Malaysia (e-mail: michael.goh@mmu.edu.my).

International Journal on Robotics, Automation and Sciences (2025) 7, 1:35-46
https://doi.org/10.33093/ijoras.2025.7.1.5
Manuscript received: 14 Sep 2024 | Revised:4 Dec 2024 | Accepted: 11 Dec 2024 | Published: 31 Mar 2025
© Universiti Telekom Sdn Bhd.
Published by MMU PRESS. URL: http://journals.mmupress.com/ijoras
This article is licensed under the Creative Commons BY-NC-ND 4.0 International License

require the body and not the arms to move are more often confused by the models

II. LITERATURE REVIEW

In the process of monitoring ADLs, there are two common ways it is done which is by cameras and sensors. Cameras were chosen for this paper due to its ability to provide visual data which has more accurate ADL monitoring.

We also investigated recent works in action recognition using conventional methods [1-5], and deep learning methods [6-10, 18-20]. A hidden Markov model approach was used for real-time activity classification using signals from wearable wireless sensor networks [1]. The focus of this study was to create an algorithm that can conduct feature analysis. The ADLs were monitored through the data contained in the sensor network. A different study used a support vector machine to classify ADLs in health smart homes [2]. Seven actions were monitored according to the Katz activity scale and the support vector machine was chosen due to its ability to handle smaller datasets. In [3], a system for activity recognition using a multi-sensor fusion utilized the Naïve Bayes classifier. A platform utilizing Shimmer wireless sensors was used to collect information related to movement and Naïve Bayes classification was adopted as the classifier. A method was proposed in [4] to monitor ADLs of elderly through wireless sensor data. The sensors were located in two places, these being 'invisible' and non-intrusive areas. The raw sensor data collected were used to detect the related actions and general human movement. Lastly, a system to measure an individual's limb's range of motion using multiple Inertial Measurement Units (IMU) was created in [5]. The acquired data was displayed and visualized in a 3D visual model by the custom program. Multiple Inertial Measurement Units sensors was used to measure certain angles at 10°, 30°, 60° and 90° to test its accuracy.

Once deep learning methods started to be more popular in the 21st century, they started to get widespread use in the monitoring of ADLs. In [6], two deep learning approach for the detection of ADLs based on Long Short-Term Memory (LSTM) networks were proposed. The first was MT-LSTM and the second was CNN-LSTM. The use of recurrent neural networks (RNN) was proposed in [7] to address the problem of classifying ADLs. The RNN model was very good at monitoring successive inputs of data but had issues with long term dependencies. In [8], a novel method for detecting and recognising ADLs using a combination of LSTM and CNN was proposed. Individuals wore a fitbit and the recorded signals was used to train and test the proposed model. The wearable accessory was embedded with the models script which allowed it to read any type of data from the sensor. In [9], a research to study recognition of human activity with CNN and RNN using smartphone and smartwatch sensors was conducted. The study explored the capability of nine different deep learning models based on their ability for action detection. These models were all based on CNNS and RNNS and were analyzed in 3 different situations using a big set of sensor data collected from smartphones, smartwatch and a combination of both devices. Lastly, A study to suggest a deep learning-based automated fall detection solution using CNN was carried out in [10]. The study used real-time video analytics and does not require the patient to put on any kind of wearable device. Table 1 shows a summary of the studied methods.

TABLE 1. State-of-the-art methods.

Author	Method	Accuracy	Pros	Cons
J. He, H. Li and J. Tan (2007)	Hidden Markov Model (HMM)	95.82%	Low data transmission requirement.	High cost, sensor has limited battery, require high computing resources.
A. Fleury, N. Noury and M. Vacher (2009)	Support Vector Machine (SVM)	Polynomial: 75.86% Gaussian: 86.21%	Low cost, good global recognition rate.	Low accuracy on different posture
L. Gao, A. K. Bourke and J. Nelson (2011)	Multi-sensor Fusion	4 Sensors: 97.66%, 3 Sensors: 92.55%, 2 Sensors: 86.29%, 1 Sensors: 78.22%	Able to tolerate network issues, still functions even when 1 sensor is offline.	High cost
Q. Zhang, M. Karunanithi, D. Bradford and Y. van Kasteren (2014)	Wireless Sensor data	Precision Rate: 82% Recall Rate: 78%	Learning model shows good potential for further improvements.	Average accuracy. high cost to install and maintain
Chen, C. H., Gan, K. B., & Abd Aziz, N. A. (2022).	Wearable Inertial Measurement Units.	IMU sensors coefficient: 0.9967	Does not require visual data.	Different individuals may have different wrist contortion angles.
G. Ercolano, D. Riccio and S. Rossi (2017)	Multi-scale LSTM, CNN-LSTM.	CNN-LSTM: Precision rate: 95.40% Recall rate: 94.38% MT-LSTM: Precision rate: 93.30% Recall rate: 92.40%	Able to detect small-motion sequences well.	Dataset used is too small.
R. Jurca, T. Cioara, I. Anghel, M. Antal, C. Pop and D. Moldovan (2018)	Recurrent Neural Network (RNN).	Basic cross validation: 82.5% Leave one subject out: 87.16%	Able to keep track of successive sensor data inputs..	High cost to setup and maintain.
P. Vanijkachorn and P. Visutsak (2021)	Deep Convolutional Long Short Term Memory (LSTM).	88.425%	High accuracy for activity classification.	Unwanted frequency noises frequently occur.
S. Mekruksavanich, P. Jantawong, N. Hnoohom and A. Jitpattanakul (2022)	CNN and RNN-based Networks using Smartphone and Smartwatch sensors.	CNN: 91.05% LSTM: 95.81% BiLSTM: 96.38% GRU: 96.36% BiGRU: 96.50% CNN-LSTM: 95.70% CNN-BiLSTM: 95.69%	BiGRU produced high accuracy with both smartphone and smartwatch data.	Dataset used is limited due to it not being hand-orientated.

		CNN-GRU: 94.86% CNN-BiGRU: 95.42%		
S. Vyshali and S. Raja Mohamed (2023)	Convolutional neural network (CNN) using real-time video analytics.	Recall rate: 90.33% Precision rate: 93.45%	High accuracy, low error rate and low false positive alarms. environments	High cost to setup and maintain.

III. PROPOSED SOLUTION

In this paper, we have identified five of the most common activities generally performed by elderly people which are eating, cooking, sweeping, standing/sitting and walking. The dataset collected for this paper has 5 different ADLs performed by 20 individuals consisting of 10 male and 10 females. Each chosen individual is in good health and are able to perform each action independently without assistance. None of the individuals require the use of walking canes or wheelchairs. A short overview of ADL video and each individual is shown in Table 2.

TABLE 2. ADL video information per person.

ADL	No of videos	Video Information
Eating	2	1. Front view 2. Side view
Cooking	2	1. Front view 2. Side view
Sweeping	2	1. Front view 2. Side view
Standing/ Sitting	4	1. Front view (standing up) 2. Front view (sitting down) 3. Side view (standing up) 4. Side view (sitting down)
Walking	4	1. Front view (right foot first) 2. Front view (left foot first) 3. Side view (right foot first) 4. Side view (left foot first)

Each ADL was taken at 2 different angles which are a front view and a side view. The movements of certain actions were split into 2 parts for more clarity. This includes the walking action which was split into walking with the right leg first and left leg first, and standing and sitting action which was split into standing and sitting separately. However, the standing sitting ADL will still be considered as a single ADL due to the similar postural transitions between these two actions. A phone camera was used to capture each action. The device used was a Redmi Note 12 Pro 5G released by Xiaomi in November of 2022. It comes with a 108 megapixel main camera which allows each video to be recorded in immense detail and clarity. Each recording was then edited into 1280x720 pixels or 720p resolution to achieve high display resolution while not making the file size too big.

The participants had to first sign a consent form before being recorded to ensure they have agreed to allow their data to be collected willingly. The ADLs were generally recorded in an indoor setting, most commonly in the living room or the kitchen. The reason behind creating our own personal dataset rather than using a public dataset is due to a few factors. Public datasets often lack the freedom and customization of angles, environment, individual movements and more. By using a personal dataset, we can capture specific actions and movements that aligns closely with our research. Additionally, as this study specifically caters towards elderly individuals, 15 out of 20 individuals were above the age of 60 with the general age range of participants going from 22 to 88. A sample of the dataset collected for the eating ADL at both front and side view is shown in Figure 1.

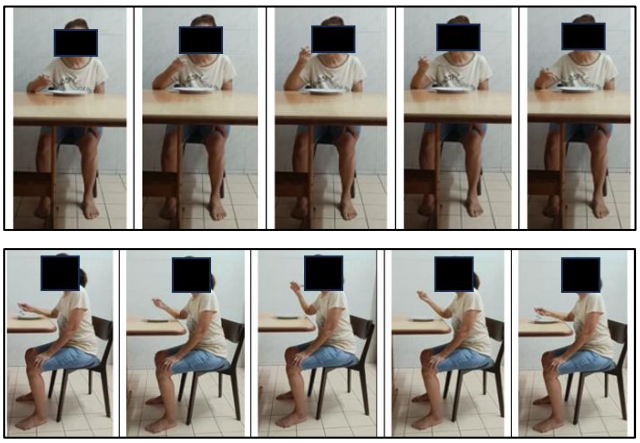


FIGURE 1. Sample of the collected dataset for the eating ADL.

A. Human Pose Estimation

To track the human pose and movement from the dataset, a program called Alphapose [11] was used. Alphapose is a tracking and estimation system that tracks whole-body human poses. PyTorch and MXNet were used as the base for its development. Alphapose supports various inputs such as image files, video files, and stream input from camera. Figure 2 illustrates the architecture of Alphapose system [11].

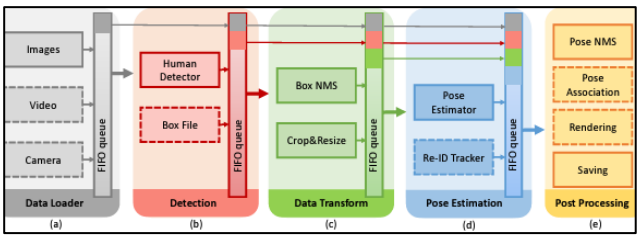


FIGURE 2. System architecture of alphapose [11].

Alphapose introduces a tracking method that is capable of tracking multiple people at once. The person re-ID is attached to the estimator in charge of poses. The Person re-ID will be able to identity any similar individuals from multiple human proposals. Due to Alphapose using the top-down framework, it extracts the re-ID features produced by every bounding box that the object detector has produced. K heatmaps will be generated where k = number of keypoints per person. PGA will then transform the heatmap produced into an attention map (mA). Due to mA having the same size as the re-ID feature map (mid), it

can collect the weighted re-ID feature map (mwid). Hadamard product is depicted by ⊙.

$$mwid=mid \odot mA+mid \quad (1)$$

In the current version, Alphapose uses detectors such as YOLOV3 and even the EfficientDet that has been trained on the COCO dataset [16]. The COCO dataset is the human keypoints prediction standard benchmark. Seventeen human body keypoints are contained in it. There are 118000 images that can be used for training, 5000 for validation and 41000 to test in total. A new backbone called FastPose [17] was designed for the pose estimator, which yields both high efficiency and accuracy. When compared to other methods such as OpenPose and Detectron, Alphapose produced the highest performance and accuracy [11].

B. Data Preprocessing

Once dataset of initial actions has been collected, it is then processed through AlphaPose to produce both a MP4 file and a json file. The json file saves the result for all images in the video and is similar to the results format used by COCO. Each json file output has 17 keypoints as well as the confidence score in the range of [1,0]. The number of outputs in the json file will depend on the number of frames in the video as each frame will produce one image. Therefore, the longer the video, the bigger the output. The json file is processed to only keep the x and y values of each keypoints and any unnecessary data is removed from the dataframe to ensure higher efficiency and accuracy.

The number of frames for each video is limited to 30 for ease of implementation. Therefore, each video will have 29 rows as the dataframe starts from row 0. All the frames are then moved into the same row. This means that 1 row will have all 30 frames with 17 keypoints each. This will result in each action having 1 row and 493 columns. Lastly an action column for the video is added to the specific dataset. As stated earlier, the walking action and standing sitting action are split into two videos each. Therefore, each subject will have 7 videos per view, making it 14 videos per subject as we have a front view and a side view. Since there are 20 subjects with 14 videos each, we will multiply 20 and 14 to get a total of 280 rows. The 493 keypoint columns will be added with the action column to create 494 columns per action. Therefore, the final dataset has 280 rows and 494 columns.

C. Feature Extraction and Classification

We will be investigating the effectiveness of 5 different deep learning models for ADL recognition. The chosen models are LSTM, Bi-LSTM, CNN, RNN and transformer model. These models were chosen based on accessibility, ease of usage and performance. We aim to analyse and determine which model performs the best at identifying the collected ADLs.

Long Short-Memory Networks (LSTM)

Long Short-Term Memory Networks [12] is a form of deep learning, sequential network that has the ability to allow information to persist. The LSTM network architecture generally consists of 3 sections. The starting cell will choose which data should be remembered from the previous timestamp. The second cell will learn new data. The last cell will pass any new data in the current timestamp onwards. All 3 cells of the LSTM units are called gates. The flow of data and data are controlled by these gates. In general, LSTM is widely used for tasks that involve sequences such as video and action recognition, speech recognition, language processing and time-series prediction.

Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNN) [13] is a type of deep learning neural networks that is generally used to analyze visual imagery. A technique called Convolution is used where a mathematical operation which operates on two functions produces a third function. A CNN is made up of four parts which is the convolutional layer, Rectified Linear Unit, pooling layers and fully connected layers. The first block of a CNN is the convolutional layer which is the main mathematical task that performs convolution. In this layer, multiple equal size filters and applied which is each used as pattern recognition for an image. The ReLu function is then applied to every convolution operation. This will assist the network in learning any non-linear relationships between image features. This will make the network more robust when identifying various kinds of patterns as well as mitigate the vanishing gradient problems. Next is the pooling layer which has the main goal of pulling any significant features from the previously created convoluted matrix in the convolution layer. Additionally, pooling also helps mitigate overfitting. The fully connected layers are the final layer of the CNN. These inputs will all correspond together into a flattened one-dimensional matrix. Lastly, a softmax prediction layer is applied to generate all probability values for every single possible outcome that is related to the output labels. The chosen outcome will have the highest probability score.

Recurrent Neural Network (RNN)

Recurrent Neural Networks (RNN) [14] is a variant of neural networks. It is unique from others as each output from the previous step acts as the input to the current step. This makes it unique compared to traditional neural networks. The main ability of the RNN is the hidden state. This state remembers and keeps past information about a sequence. It will use the same parameters for both input and hidden layers. This will help reduce the parameter complexity. A fundamental unit in a RNN is the recurrent unit. This unit is able to keep a hidden state which will assist the RNN network in capturing sequential dependencies. Generally, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) will improve on RNNs capability to handle and process any long-term dependencies. This makes RNNs useful when it comes to learning time series prediction tasks due to its feature to remember past inputs.

Bidirectional LSTM (Bi-LTSM)

Bi-LSTM is a RNN that processes sequential data both in forward as well as backward directions. In other words, it combines the power of bidirectional

processing and LSTM that enables the model to collect and analyse past and future information of any input sequence. Bi-LSTM is made up of two LSTM layers that will perform sequence processing in a forward direction and a backward direction. Figure 3 shows the Bi-LSTM architecture.

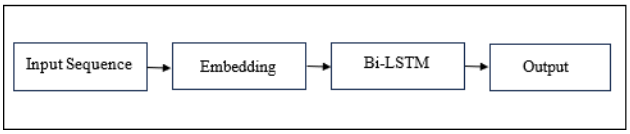


FIGURE 3. Bi-LSTM architecture.

The input sequences consist of data point sequence which is usually represented as a vector. This sequence is then embedded and transformed into a dense vector representation. This embedding assist in capturing the sematic meaning of data points which will help give a more meaningful and compact representation of any subsequent layers. The Bi-LSTM layer will then process it in a forward and backward direction simultaneously with its own set of parameters. The output is generated by the combination of the hidden states in the LSTM layers at every time step. Generally, Bi-LSTMs are used to collect and analyse long-term dependencies in certain data that is sequential as it is able to process the information in both forward and backward directions. This makes it suitable for tasks that require modelling context over a very long period of time such as language processing and speech recognition.

Transformer model

Transformer model [15] is a neural network that was created primarily for language and text processing. Therefore, a big part of its architecture is the transformation of text input into numerical representation. This is called embedding and it works by transforming each word input into a vector of high dimension. The elements where the text is divided into to perform embedding is called tokens. Unliked RNNs, the transformer is not able to remember how input sequences were fed into the model. This gives it a few challenges such as limited context dependency. This happens because it is not able to keep long-term dependency information and is also not able to correlate words that appeared several time steps ago. Additionally, it also suffers from context fragmentation as the model is trained from scratch at every segment which leads to performance issues.

Implementation details

Before the dataset can be used in a deep learning models, there are a few modifications to be done to prepare it. Firstly, the values in keypoints columns which are currently tuples with object datatype are converted to a numpy array with float datatype. The sum of both the x and y value is calculated and placed in each keypoint. This is due to certain models not being able to process object datatypes. Since certain models are only designed to handle numerical data, the action values will also need to be converted. A label map is created for each action with a number assigned to each of them. From the label map, the to_categorical function from tensorflow library is used to further convert it to a binary class matrix. This will result in a binary class matrix being formed and representing each action as shown below. The created binary matrix has a size of 140x12 where each row represents the action associated with it. The dataset is then split into training and testing sets. The train_test_split function from sklearn library is used to split the data into training set and testing set with a testing size of 0.25.

IV. EXPERIMENTAL RESULTS

A. Human Tracking Results

Alphapose is used to process the collected dataset to produce the human tracking results. There are two views for each action, front and side. Figure 4 shows the video output for the eating ADL in both front and side view once it has been processed by Alphapose.

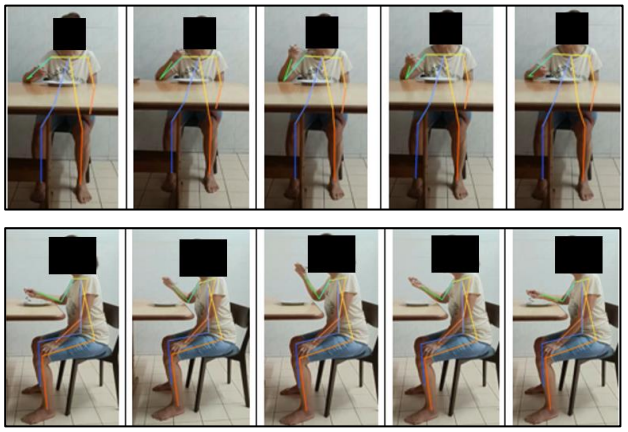


FIGURE 4. Sample of the collected dataset of the processed eating ADL.

Generally, the tracking results produced by Alphapose were very accurate. This was determined by evaluating the rgb skeleton produced which represents the keypoints of the individual. This can be attributed to a few factors. The first being the experimental setup that aimed to make the human tracking as clear as possible. This included only having a single person setting per video and very bright lighting. The background was kept as simple as possible with it only being a white wall and the recorded individual had all their body parts clearly shown to the camera to prevent any tracking issues from occurring. However, there were a few cases where the results were slightly off due to a certain body part of the subject being blocked by an object. For example, Figure 5 shows a human tracking result that is unable to render the rgb skeleton correctly.



FIGURE 5. Sample of incorrect human tracking result for the cooking ADL.

This error might have occurred due to the right leg of the subject being obstructed by the table. Thus, Alphapose might have not been able to recognise it as the human leg which causes the output to not be accurate. This inaccuracy can be overcome by recapturing the action without having any object interfering with the subject.

B. Action recognition results

Long Short-Term Memory (LSTM)

The first model is the LSTM model. The created LSTM model is then fitted with the parameter epochs=250 with a validation split of 0.1. This means the model runs through the dataset 250 times. 10% of the training data is then used as validation set while the remaining 90% will be used to train the LSTM model. Once the model has finished training, it managed to achieve a mean squared error of 2.58% and a categorical accuracy of 80.00% on the testing set as shown in Table 3.

TABLE 3. LSTM results.

Categorical Accuracy	Mean squared error (MSE)
0.80	0.0258

From the classification report in Table 4, a few observations can be made. The prediction for label 2 (stand_side) performed badly while labels 8 (eat_front), 9 (eat_side) and 10 (cook_front) performed the best. In general, the standing and sitting ADL side view did not perform well. This may be due to the short number of frames used which cause the standing and sitting side to be confused with similar side actions such as sweeping side view. In comparison, the eating and cooking ADL performed the best. During recording, all the individuals tend to move their hands in the cooking and eating motion in a similar manner as it was shown to them before recording. Therefore, the model would have had an easy time predicting this ADL. Overall, the LSTM model performed generally well with an accuracy of 80.00%.

TABLE 4. LSTM classification report.

Label	Precision	Recall	F1-score	Support
0	1.00	0.40	0.57	5
1	1.00	0.75	0.86	8
2	0.00	0.00	0.00	4
3	0.75	0.60	0.67	5
4	1.00	0.67	0.80	3
5	0.75	0.60	0.67	5
6	0.64	1.00	0.78	7
7	0.75	1.00	0.86	9
8	1.00	1.00	1.00	3
9	1.00	1.00	1.00	5
10	1.00	1.00	1.00	4
11	0.86	1.00	0.92	12
Accuracy			0.80	70
Macro Avg	0.81	0.75	0.76	70
Weighted Avg	0.81	0.80	0.79	70

Figure 6 shows the confusion matrix plotted with true labels and predicted labels for each action. The cook_side and walk_side ADL performed the best achieving a score of 12 and 9 respectively. The stand_side ADL seem to be wrongly classified the most, achieving zero true label and three inaccurate labels.

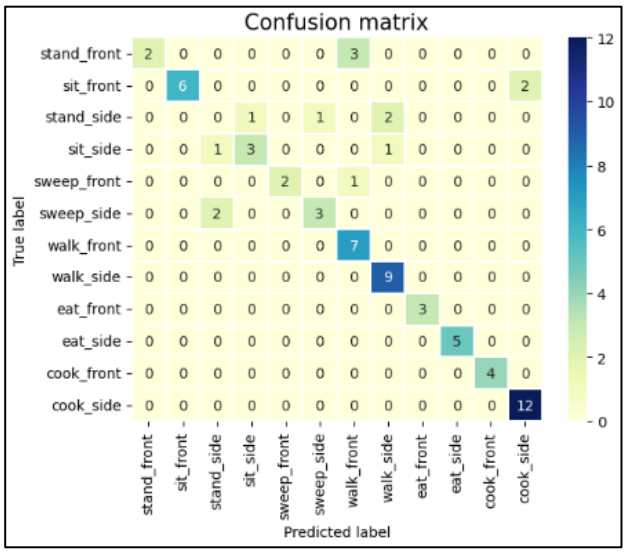


FIGURE 6. LSTM confusion matrix.

Figure 7 shows the training and validation loss graph that is plotted to view the trend of data loss in both training set and validation set. From the graph above, it can be viewed that both training and validation loss experience big spikes at similar intervals. This could be due to suboptimal hyperparameters on the model. However, both training data loss and validation loss continues on a decreasing trend throughout the 250 epochs.

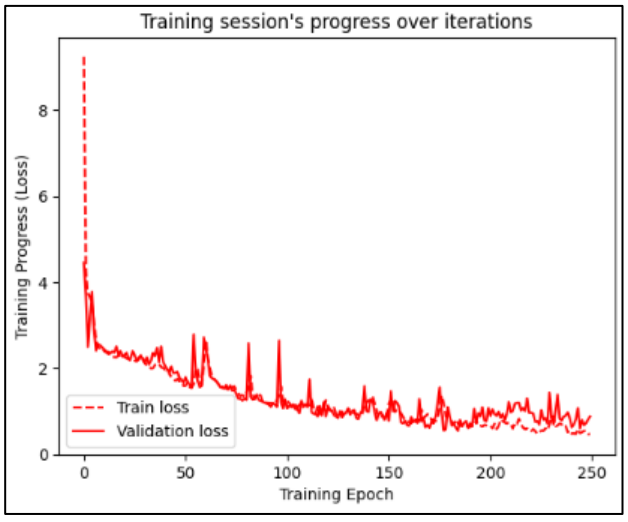


FIGURE 7. LSTM loss graph.

Bidirectional LSTM (Bi-LSTM)

The next model is Bi-LSTM. Similar to LSTM, it is also fitted with parameters epoch = 250 and a validation split of 0.1. An extra parameter batch_size =

128 is also added to improve the model's performance. Once the model finished training, it managed to achieve a categorical accuracy of 65.71% and a mean squared error of 3.96% on the testing set as shown in Table 5.

TABLE 5. Bi-LSTM results.

Categorical Accuracy	Mean squared error (MSE)
0.6571	0.03956

From the classification report generated in Table 6, it is observed that label 1 (sit_front), label 2 (stand_side) and label 4 (sweep_front) performed the worst. This may have happened due to the feature overlap between actions such as sitting front and sweeping front. The limitations of 30 frames should play a big role in this issue as the duration may not be long enough for the bi-lstm model to adequately capture the unique movements of each action.

Label 7 (walk_side), label 8 (eat_front), label 9 (eat_side) and label 11 (cook_side) performed well and each achieved a f1-score of above 0.80. The eating ADL performed well in this model which might have been due to the uniqueness of this action compared to others.

TABLE 6. Bi-LSTM results.

Label	Precision	Recall	F1-score	Support
0	0.60	0.75	0.67	4
1	0.00	0.00	0.00	2
2	0.17	0.50	0.25	2
3	0.80	0.50	0.62	8
4	0.67	0.22	0.33	9
5	1.00	0.67	0.80	6
6	0.71	0.83	0.77	12
7	0.88	0.88	0.88	8
8	1.00	0.75	0.86	8
9	0.75	1.00	0.86	3
10	0.60	0.60	0.60	5
11	0.00	1.00	1.00	3
Accuracy			0.66	70
Macro Avg	0.68	0.64	0.64	70
Weighted Avg	0.76	0.66	0.68	70

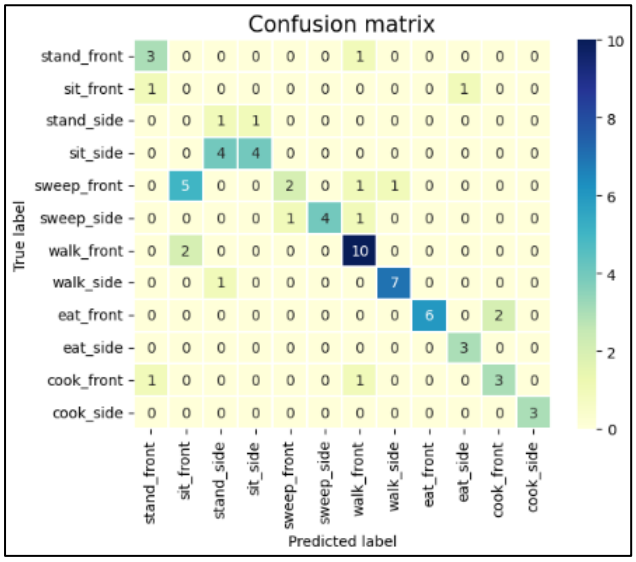


FIGURE 8. Bi-LSTM confusion matrix.

Figure 8 shows the plotted confusion matrix and it can be observed that the sit_front ADL was wrongly classified as sweep_front quite often. This is due to both actions having similar movements and angle such as both requiring the subject to bend lower and face the camera. Similarly, the cook_front ADL is also wrongly classified as the eat_front ADL. This may have been due to both actions requiring similar movements and motions of the subject's hands.

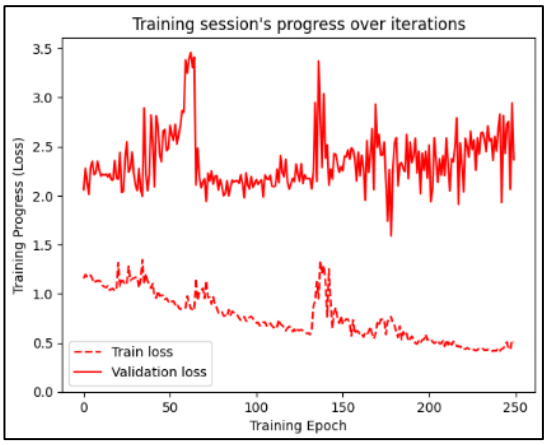


FIGURE 9. Bi-LSTM loss graph.

Figure 9 shows the training and validation loss graph. It is observed that the validation loss has huge spikes at irregular intervals and is generally on an increasing trend. The training loss also has spikes at certain intervals but is generally on a decreasing trend.

Convolutional Neural Network (CNN)

The CNN model is fitted with the parameter epoch=50 and batch_size = 32. The number of epochs is decreased from the LSTM models previously due to the CNN reaching high accuracy quickly and to prevent the model from overfitting. Once training was finished, it was determined that the CNN model achieved a categorical accuracy of 82.86% and a mean squared error of 1.91% as shown in Table 7.

TABLE 7. Bi-LSTM results.

Categorical Accuracy	Mean squared error (MSE)
0.8286	0.01914

The classification report for CNN is generated in Table 8 and shows that majority of ADLs perform well with a few outliers. Label 2 (stand_side), label 3 (sit_side) and label 4 (sweep_front) performed badly with stand_side performing the worst achieving an F1-score of 0.00. Similar to both LSTM models discussed previously, the standing side ADL seem to be performing badly and is constantly being wrongly classified. This can be due to the action not having much uniqueness in the motions of arms and only requires the subject's torso and legs to move up. The rest of the ADLs all performed very well achieving a F1-score of above 0.80.

TABLE 8. Bi-LSTM results.

Label	Precision	Recall	F1-score	Support
0	1.00	1.00	1.00	5
1	1.00	0.75	0.86	4
2	0.00	0.00	0.00	4
3	0.43	0.50	0.46	6
4	0.50	1.00	0.67	2
5	1.00	1.00	1.00	7
6	1.00	0.92	0.96	13
7	0.91	0.83	0.87	12
8	1.00	1.00	1.00	4
9	1.00	0.80	0.89	5
10	0.75	1.00	0.86	3
11	1.00	1.00	1.00	5
Accuracy			0.83	70
Macro Avg	0.80	0.82	0.80	70
Weighted Avg	0.85	0.83	0.84	70

Figure 10 shows the confusion matrix for CNN model. It is observed that the stand_side ADL is always being wrongly classified as the sit_side ADL. The sit_side ADL is also frequently being wrongly classified as the stand_side ADL. This may be due to all of these actions being the same angle and not requiring any unique motions of the subject's arms. Thus, the CNN model is unable to properly capture and extract the correct features which causes it to confuse actions of similar movements.

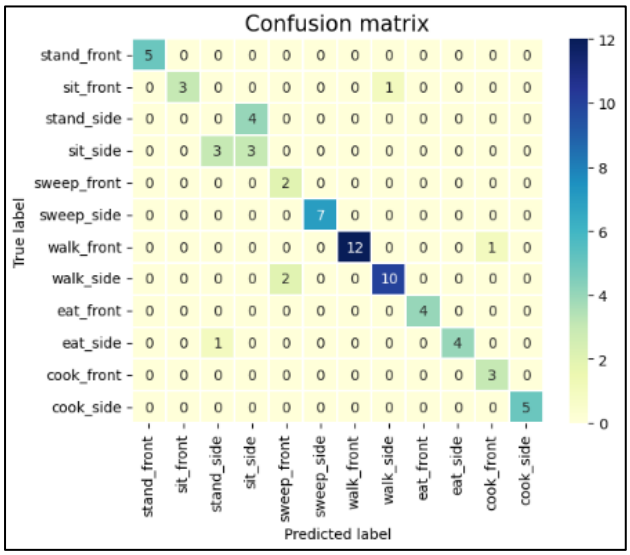


FIGURE 10. CNN confusion matrix.

Figure 11 shows the training and validation loss graph for CNN. It is observed that both losses decrease drastically in the beginning signifying that the CNN model learns very rapidly. However, this can also be a sign that the model is potentially overfitting. This is not surprisingly as CNN models are generally prone to overfitting, especially when trained on small datasets such as the current ADL dataset.

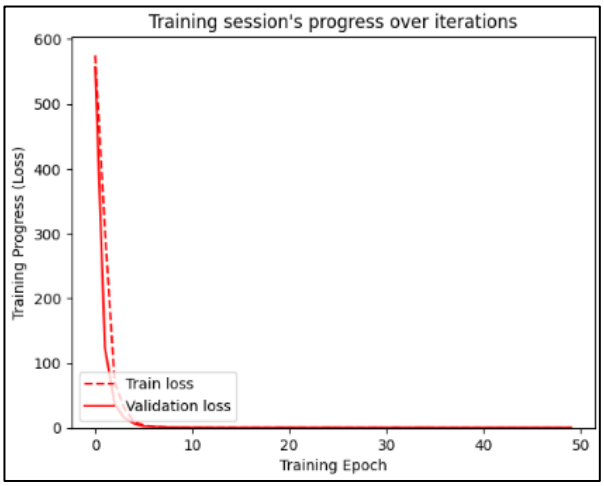


FIGURE 11. CNN loss graph.

Recurrent Neural Networks (RNN)

The RNN model is fitted with the parameters epoch = 80 and batch_size = 16. Once training was completed, the model managed to achieve a categorical accuracy of 67.14% and mean squared error of 3.97% on the testing set as shown in Table 9.

TABLE 9. RNN results.

Categorical Accuracy	Mean squared error (MSE)
0.6714	0.03974

The classification report for the RNN model is generated in Table 10 and shows that label 1 (sit_front), label 2 (stand_side) and label 3 (sit_side) performed poorly. Similar to previous models discussed, these actions do not have much uniqueness in terms of movements and motions of the arms and only requires the body and legs to move up and down. This would have caused the model to not capture and extract the proper features and confuse it with other actions of similar body movements. On the other hand, label 8 (eat_front), label 9 (eat_side) and label 11 (cook_side) which all requires unique motions of the arms and hands of the subject performed well achieving a score of 1.00 each.

TABLE 10. RNN classification report.

Label	Precision	Recall	F1-score	Support
0	0.40	0.86	0.55	7
1	0.50	0.25	0.33	4
2	0.43	0.38	0.40	8
3	0.33	0.20	0.25	5
4	0.57	0.50	0.53	8
5	0.33	1.00	0.50	1
6	0.90	0.82	0.86	11
7	0.88	0.78	0.82	9
8	1.00	1.00	1.00	2
9	1.00	1.00	1.00	4
10	1.00	0.67	0.80	6
11	1.00	1.00	1.00	5
Accuracy			0.67	70
Macro Avg	0.70	0.70	0.67	70
Weighted Avg	0.71	0.67	0.67	70

Figure 12 shows the confusion matrix for the RNN model. It is observed that the stand_front ADL has the highest number of wrong classifications with it being wrongly classified by the RNN as sit_front, sweep_front and walk_front. This may be due to all of these actions being the same angle and also requiring the user to slightly bend their body as in the case of sitting and sweeping.

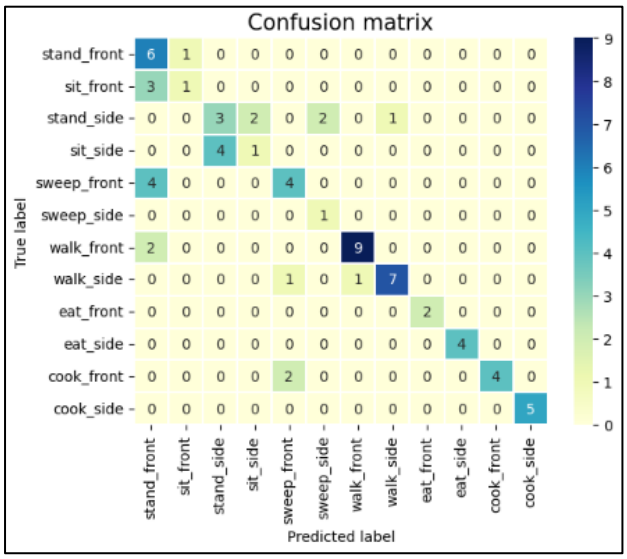


FIGURE 12. RNN confusion matrix.

Figure 13 shows the training and validation loss graph for the RNN model and the results appear to be quite similar to the CNN model. Both training and validation loss drastically drops and lingers at a very low range for the remainder of training epochs. This can also be indicative that the model is experiencing overfitting.



FIGURE 13. RNN loss graph.

Transformer model

The last model is the transformer model and it is fitted with the parameters epoch = 50 and batch_size = 16. After the model finished training, it managed to achieve a categorical accuracy of 60.00% and mean squared error of 3.62% as shown in Table 11.

TABLE 11. Transformer results.

Categorical Accuracy	Mean squared error (MSE)
0.60	0.03623

The classification report for the transformer model is generated in Table 12 and shows that label 1 (sit_front) and label 4 (sweep_front) performed poorly with a score of 0.00 each. In general, the standing and sitting ADL for the transformer model performed terribly with both front and side angles of the action not crossing 0.40 F1-score. The transformer model excels at capturing long-range dependencies but is not as good at capturing short sequences. This issue is highlighted even more when the actions do not have much distinguishing features such as arm movement in the case of standing and sitting ADL. Actions that have more distinct features such as label 9 (eat_side) and label 11 (cook_side) tend to performed better as they have a wide range of arm movements that distinguishes it from the rest.

TABLE 12. Transformer classification report.

Label	Precision	Recall	F1-score	Support
0	0.22	0.50	0.31	4
1	0.00	0.00	0.00	5
2	0.25	0.20	0.22	5
3	0.43	0.33	0.38	9
4	0.00	0.00	0.00	3
5	1.00	0.62	0.77	8
6	0.75	0.86	0.80	7
7	0.57	1.00	0.73	8
8	0.60	0.75	0.67	4
9	1.00	1.00	1.00	5
10	1.00	0.40	0.57	5
11	1.00	1.00	1.00	7
Accuracy			0.60	70
Macro Avg	0.56	0.56	0.54	70
Weighted Avg	0.62	0.60	0.58	70

Figure 14 shows the confusion matrix for the transformer model. It is observed that the stand_front ADL is wrongly classified with other ADLs with the front angle such as sit_front, sweep_front, walk_front and cook_front. The cook_side ADL performed extremely well with it not being wrongly classified at all.

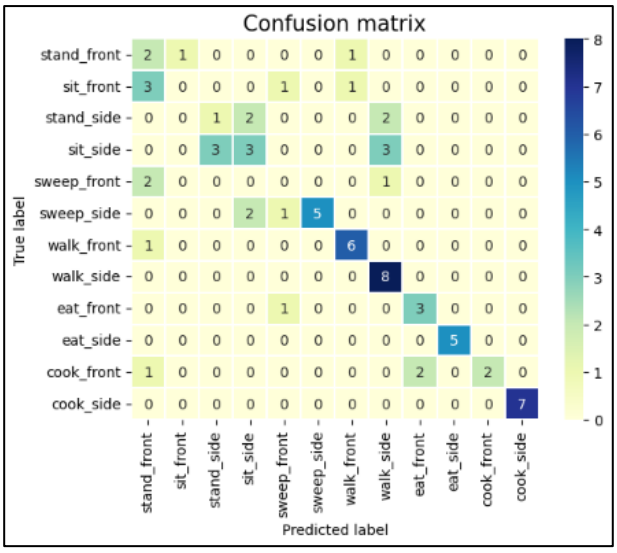


FIGURE 14. Transformer confusion matrix.

Figure 15 shows the training and validation loss graph for the transformer model. It is observed that both training and validation loss are on the downwards

trend throughout the 50 epochs. The validation loss experiences frequent minor spikes while the training loss does not and just continues decreasing.



FIGURE 15. Transformer loss graph.

Results Comparison

Table 13 presents the results achieved by each model including their accuracy, mean squared error as well as the total time taken for the model to complete training.

TABLE 13. Results comparison

Deep Learning Model	Accuracy	Total Time Taken	Mean Squared Error
LSTM	0.8000	48 seconds	0.02580
Bi-LSTM	0.6571	46 seconds	0.03956
RNN	0.6714	388 seconds	0.03974
CNN	0.8286	14 seconds	0.01914
Transformer model	0.6000	210 seconds	0.03622

From the results in Table 13, various observations can be made. Firstly, the LSTM model was able to achieve a high accuracy of 80.00% with decent training speed at 48 seconds. This shows that the LSTM model is very effective in learning temporal dependencies and long-term patterns such as the ADLs in the dataset. It is also efficient as it trains relatively quickly. Therefore, the LSTM model provides a good balance between accuracy and training speed which makes it very suitable for ADL recognition. The Bi-LSTM model underperforms when compared to the standard LSTM model achieving only an accuracy of 65.71% with a training speed of 46 seconds. The Bi-LSTM model not performing well might be due to certain ADLs not really requiring bidirectional context. Additionally, Bi-LSTM requires very delicate tuning of the model's layers and hyperparameters which might not have been set to its utmost optimal value when training on the dataset.

The RNN model has an accuracy of 67.14%. This is lower than the standard LSTM which is to be expected as RNN models generally struggle with long-term dependencies due to issues such as vanishing gradients. It also has the longest training time out of all the models by far at 388 seconds. This long training time makes the model inefficient at handling ADL recognition tasks when compared to LSTM and Bi-LSTM both in terms of accuracy and training time. The CNN model achieves the highest accuracy and fastest training speed at 82.86% and 14 seconds. This indicates that the CNN model is very effective at capturing special patterns in data which is important in ADL recognition tasks. This makes the CNN model very efficient in terms of both training time and computational resources.

The transformer model has the lowest accuracy at 60.00% and a training speed of 210 seconds. Transformers are very good at capturing long-range dependencies but require a high amount of data and fine-tuning in order to perform well in ADL recognition tasks. This is because the transformer model was created primarily for language processing tasks which require a big amount of training data. The current dataset may not have enough data for it to perform to its utmost capabilities. Transformer models also require very specific and careful tuning of hyperparameters. It may have been possible that the current hyperparameters are not the best which led to suboptimal performance.

TABLE 14. ADL accuracy percentage.

ADL	Accuracy Percentage
Stand_front	0.50
Sit_front	0.79
Stand_side	0.50
Sit_side	0.54
Sweep_front	0.53
Sweep_side	0.87
Walk_front	0.81
Walk_side	0.77
Eat_front	0.90
Eat_side	1.00
Cook_front	0.84
Cook_side	0.94

From Table 14, we can observe that ADLs that require more movement and motions of the arms and legs such as eating, cooking and walking are easier for the models to detect and recognise. ADLs that have less arm and leg movement such as standing and sitting are more often confused with other ADLs that have similar body movement. Sweeping action has interesting results as the side view performed well while the front view did not. This might be due to the front view having less variation in their coordinates which makes it difficult for the models to differentiate it from other actions with similar body posture such as standing and walking. The sweeping side view allows for more distinct movement patterns of the keypoints, especially in the arms which makes it more identifiable. Additionally, viewing the sweep action from the front might exhibit more symmetry which leads to redundant keypoint information. This redundancy can cause the models to have issues distinguishing between actions that have symmetrical movements.

In summary, the CNN model is the best performer both in terms of accuracy and speed at 82.86% and 14 seconds. This is largely due to its strong capabilities at capturing spatial features. The runner-up is the LSTM model with an accuracy of 80.00% and reasonable training speed of 48 seconds. Both Bi-LSTM and RNN models have lower accuracy and longer training times compared to the CNN and LSTMs which make them less suitable for ADL recognition tasks. This is highly evident in the case of RNN which has the longest training speed at 388 seconds. Lastly, the transformer model has the lowest accuracy at 60.00% but shows slight potential and requires further optimization and fine-tuning in order to reach its fullest potential.

V. CONCLUSION

The rise in elderly individuals living alone has increased the risk of undetected health issues, making monitoring their ADLs an effective method for identifying early signs of illness through changes in movement. Five common ADLs were chosen for this research and performed by 20 individuals. The collected dataset was then trained on multiple other neural network models such as CNN, RNN, Bi-LSTM and Transformer model. It was then determined that the CNN model performed the best with an accuracy of 82.86% due to its strong capabilities at capturing spatial features. In the future, we intend to further increase the size of the dataset and number of frames for each ADL. This will drastically increase the amount of data and information that can be fed into the models and improve the results. Once the model is determined to be good enough, it can be utilized in real-time monitoring of elderly individuals. Whenever the model detects that the elderly individual is performing ADLs in a weird or unusual manner it will relay the information to healthcare services in the area. We also intend to test the model's performance under more complex environments to observe if it can be usable for real-life situations.

ACKNOWLEDGMENT

We thank the anonymous reviewers for the careful review of our manuscript.

FUNDING STATEMENT

This research is supported by the Fundamental Research Grant Scheme (MMUE/190222).

AUTHOR CONTRIBUTIONS

Josiah Wey Tsen Lim: Conceptualization, Data Curation, Methodology, Validation, Writing – Original Draft Preparation;

Tee Connie: Project Administration, Writing – Review & Editing;

Michael Kah Ong Goh: Project Administration, Supervision, Writing – Review & Editing.

CONFLICT OF INTERESTS

No conflict of interests were disclosed.

ETHICS STATEMENTS

Our research work follows The Committee of Publication Ethics (COPE) guideline. https://publicationethics.org.

REFERENCES

[1] J. He, H. Li and J. Tan, "Real-time Daily Activity Classification with Wireless Sensor Networks using Hidden Markov Model," *Proceedings of 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 3192-3195, 2007.
DOI: https://doi.org/10.1109/IEMBS.2007.4353008

[2] A. Fleury, N. Noury and M. Vacher, "Supervised classification of activities of daily living in health smart homes using SVM," *Proceedings of Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 6099-6102, 2009.
DOI: https://doi.org/10.1109/IEMBS.2009.5334931

[3] L. Gao, A.K. Bourke and J. Nelson, "A system for activity recognition using multi-sensor fusion," *Proceedings of Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 7869-7872, 2011.
DOI: https://doi.org/10.1109/IEMBS.2011.6091939

[4] Q. Zhang, M. Karunanithi, D. Bradford and Y. van Kasteren, "Activity of Daily Living assessment through wireless sensor data," *Proceedings of 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 1752-1755, 2014.
DOI: https://doi.org/10.1109/EMBC.2014.6943947

[5] C.H. Chen, K.B. Gan and N.A Abd Aziz, "Upper Limbs Extension and Flexion Angles Calculation and Visualization Using Two Wearable Inertial Measurement Units", *International Journal on Robotics, Automation and Sciences*, Vol. 4, pp. 1–7, 2022.
DOI: https://doi.org/10.33093/ijoras.2022.4.1

[6] G. Ercolano, D. Riccio and S. Rossi, "Two deep approaches for ADL recognition: A multi-scale LSTM and a CNN-LSTM with a 3D matrix skeleton representation," *Proceedings of 26th IEEE International Symposium on Robot and Human Interactive Communication*, pp. 877-882, 2017.
DOI: https://doi.org/10.1109/ROMAN.2017.8172406

[7] R. Jurca, T. Cioara, I. Anghel, M. Antal, C. Pop and D. Moldovan, "Activities of Daily Living Classification using Recurrent Neural Networks," *Proceedings of 17th RoEduNet Conference: Networking in Education and Research*, pp. 1-4, 2018.
DOI: https://doi.org/10.1109/ROEDUNET.2018.8514124

[8] P. Vanijkachorn and P. Visutsak, "A Deep Convolutional LSTM for ADLs Classification of the Elderly," *Proceedings of International Conference on Data Analytics for Business and Industry,* pp. 124-128, 2021. DOI: https://doi.org/10.1109/ICDABI53623.2021.9655856

[9] S. Mekruksavanich, P. Jantawong, N. Hnoohom and A. Jitpattanakul, "Heterogeneous Recognition of Human Activity with CNN and RNN-based Networks using Smartphone and Smartwatch Sensors," *Proceedings of 3rd International Conference on Big Data Analytics and Practices*, pp. 21-26, 2022.
DOI: https://doi.org/10.1109/IBDAP55587.2022.9907460

[10] S. Vyshali and S. Raja Mohamed, "Fall Detection & Daily Living Activity Recognition using CNN," *Proceedings of International Conference on Inventive Computation Technologies*, pp. 478-483, 2023.
DOI: https://doi.org/10.1109/ICICT57646.2023.10133971

[11] H.S. Fang, J.F. Li, H.Y. Tang, C. Xu, H.Y. Zhu, Y.L. Xiu, Y.L. Li and C.W. Lu, "AlphaPose: Whole-Body Regional Multi-Person Pose Estimation and Tracking in Real-Time," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 45, No. 6, pp. 7157-7173, 2023.
DOI: https://doi.org/10.1109/TPAMI.2022.3222784

[12] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Proceedings of Neural Computation*, Vol. *9, No.* 8, pp. 1735–1780, 1997.
DOI: https://doi.org/10.1162/neco.1997.9.8.1735

[13] L. Alzubaidi, J. Zhang, A.J. Humaidi, et al, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” *Journal of Big Data*, Vol. 8, pp. 53, 2021.
DOI: https://doi.org/10.1186/s40537-021-00444-8

[14] A. Sherstinsky, “Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network,” *Physica D: Nonlinear Phenomena*, Vol. 404, pp. 0167-2789, 2020.
DOI: https://doi.org/10.1016/j.physd.2019.132306

[15] T. Lin, Y. Wang, X. Liu and X. Qiu, “A survey of transformers,” *AI Open*, Vol. 3, pp. 111-132, 2022.
DOI: https://doi.org/10.1016/j.aiopen.2022.10.001

[16] T.Y. Lin, et al, “Microsoft COCO: Common Objects in Context,” *Computer Vision – ECCV 2014, Lecture Notes in Computer Science*, Vol 8693. *Springer, Cham*, 2014.
DOI: https://doi.org/10.1007/978-3-319-10602-1_48

[17] Z. Zhang, Z.W. Zou, P. Li, Y. Li, H. Su and G. Huang, “FastPose: Towards Real-time Pose Estimation and Tracking via Scale-normalized Multi-task Networks,” *arXiv preprints*, pp. 05593, 2019.
DOI: https://doi.org/10.48550/arXiv.1908.05593

[18] M. S. Z. Ahmad, N. A. Ab. Aziz, and A. K. Ghazali, "Development of automated attendance system using pretrained deep learning models," *International Journal on Robotics, Automation and Sciences*, vol. 6, no. 1, pp. 6–12, 2024. https://doi.org/10.33093/ijoras.2024.6.1.2

[19] T. J. Jie and M. S. Sayeed, "Review on detecting pneumonia in deep learning," *International Journal on Robotics, Automation and Sciences,* vol. 6, no. 1, pp. 70–77, 2024.
DOI: https://doi.org/10.33093/ijoras.2024.6.1.10

[20] H. Youns, S. Abbas, U. Hayat, M. H. Musaddiq, and A. Hashmi, "A cutting-edge hybrid approach for precise COVID-19 detection using deep learning," *International Journal on Robotics, Automation and Sciences*., vol. 6, no. 1, pp. 86–93, 2024.
DOI: https://doi.org/10.33093/ijoras.2024.6.1.12