Hazardous Human Activity Recognition in Hospital Environment Using Deep Learning

Khairunnisa' Ahmad Shahrim, Abdul Hadi Abd Rahman, Shidrokh Goudarzi

***Abstract*—Human activity recognition system is an essential requirement for a service robot to understand its environment and avoid obstacles. However, the robot safety feature requires the hazard level of the detected object to decide on necessary action. This study aims to develop a model to identify the type of human activity and determine the hazard level for the robot by using a region of interest-based decision-making approach. A total of 1900 images of the five most potentially hazardous activities in the hospital environment from a robot perspective are collected and used for training. Three deep learning models, namely, YOLOv2, VGG16, and MobileNetv2 SSD, are used to classify hazardous activity. Experimental results using the Deeplearning4J (DL4J) and TensorFlow frameworks show that the VGG16 model exhibits the highest performance with an accuracy of 93.33%. The YOLOv2 and MobileNetv2 SSD models obtain an accuracy of 46.67% and 68.95%, respectively. The misclassification of activity in the hospital environment is due to the high similarity of activities. Further study should be performed by collecting more data in different classes in the actual hospital environments.**

***Index Terms*— computer vision, deep learning, human activity recognition, DL4J, object detection, robot service**

I. Introduction

HUMAN activity recognition (HAR) is a process that determines human activity by using data sensors [1]. Human activity involves one or more movements of the human body. The basics of human action include walking, standing, moving to the right or left, and sitting. Therefore, the goal of this system is to be able to classify the same activities even when performed by different actors in different environments. Various applications, such as smartphone chips [2], video surveillance systems [3], and service robots [4], have used this system.

Recently, deep learning techniques have emerged as powerful methods for learning feature representations automatically from data [5]. In 2012, the author proposed a AlexNet is a deep convolutional neural network that achieved a record-breaking image classification accuracy in the Large-Scale Visual Recognition Challenge [6]. Subsequently, research has focused on deep learning methods in many computer vision application areas [7]. Many deep learning approaches have emerged in object detection with the bounding box, and tremendous progress has been achieved.

Studies on the HAR system in hospitals are poorly implemented by researchers. Robot services are essential due to the pandemic that hit the world recently to avoid contact between health officials and hospital patients [8]. Therefore, a robot needs to understand the surrounding conditions to operate effectively. Apart from the basics of human activity, various complex and hazardous activities [9], such as pushing an emergency trolley, pushing a wheelchair, and holding an intravenous (IV) drip, are performed by robots. If the robot cannot recognize the activity type, then it can fail to understand its environment. Robot services are often applied in the human environment. Thus, a collision between a robot and a human can occur unexpectedly if the robot does not follow the path provided. Therefore, robots need to be programmed and trained to avoid collisions [10]. The key contributions of this study are as follows;

(1) We evaluate three deep learning models for detecting hazardous human activity in the hospital environment using the DL4J library [11] and the TensorFlow framework.

(2) We present a region of interest (ROI)-based approach to identify the level of potentially hazardous human activity in the hospital environment.

II. Related work

Several techniques have been introduced to detect human activity involving object recognition using traditional to modern techniques [12]. The implementation of HAR was previously conducted by using spatial features and support vector machine (SVM) to classify the feature representations, either dense or sparse spatial points on histograms [13]. Convolutional neural network (CNN) started to gain momentum, showing that deep learning can find relevant features and outperform these methods significantly [14, 15]. The author in [16] used artificial Neural Networks (ANNs) to design an HAR system that can identify suspicious activity in an area. The other authors combined three types of classification techniques, namely, SVM, decision tree, and ANN, to identify human activity by service robots. The results obtained from the three algorithms are compared [17].

The author in [18] used channel state information (CSI) collected from multiple access points to classify four different activities [18]. A deep learning model is compared with an SVM approach, consisting of various inputs into a CNN

Manuscript received September 28, 2021; revised July 10, 2022. This work was supported by Universiti Kebangsaan Malaysia under grant code: UKM-TR-021.

Khairunnisa' Ahmad Shahram is a graduate student at the Faculty of Information Science & Technology, Universiti Kebangsaan Malaysia, Selangor, Malaysia. E-mail: A168306@siswa.ukm.edu.my

Abdul Hadi Abd Rahman is a senior lecturer at the Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, Selangor, Malaysia. (corresponding author to provide e-mail: abdulhadi@ukm.edu.my).

Shidrokh Goudarzi is a lecturer at the Centre for Vision Speech and Signal Processing (CVSSP), The University of Surrey, UK. E-mail: s.goudarzi@surrey.ac.uk

combined into a fully connected layer and then into a classifier. The results show that deep learning models the data better, achieving greater accuracy. The CSI data are transformed into spectrographs and divided into time windows. Some studies have compared long short-term memory (LSTM) to one of the most common univariate models for time series, that is, the auto-regressive integrated moving average model, by using wireless fidelity (WiFi) data [19]. LSTM predictions are significantly better, reducing the root mean square error by between 80.9% and 93.4%, showing that the deep learning approach is more promising in this area.

However, some HAR systems have been proposed in recent years [14], [15], [16], [17]. A learning framework adopted still requires an architecture that is suitable for recognizing human activity and can outperform the traditional classification methods, such as decision tree and SVM [14], [17]. We propose three deep learning approaches to address these issues and compare them with the system in recognizing human activities in hospital hallways.

III. Methodology

The model's methodology developed in this study comprises six phases; data collection, preprocessing, experimental setup, deep learning models, the ROI phase in the best model, and evaluation matrices.

A. Data collection and preprocessing

A survey was conducted in one of the hospitals in Kajang, Malaysia, to fulfill the user requirements and obtain the most common human activities in the hospital hallway. We chose five main classes based on special interviews with hospital authorities to be used in the data collection phase; hold IV drip, push wheelchair, push medical cart, standing, and walking. The datasets were collected from Google Images by using a customized Python script and Kaggle. The images included several keywords, such as human activity, hospital hallway, and hazardous activity in hospital, excluding cartoons and transparent background. The filtered data consist of 1900 images, with 300 images in three classes: hold IV drip, push wheelchair, and push medical cart, and 500 images in standing and walking. The first three classes are less than standing and walking due to a lack of dataset resources involving human hospital activities.

All the collected images were resized into 224×224 and 416×416 pixels because VGG16 [20] and YOLOv2 [21] models have default input image sizes, and image resizing was not included in DL4J. Each image was labeled in five defined classes by using an open-source tool called *LabelImg*. The dataset was preprocessed and split into 80% for the training set and 20% for the testing set. The train-test split helps in comparing our models' performance in class prediction.

B. Experimental setup

Suitable hardware specifications need to be prepared because deep learning models require high computational power to build our models. Experimental hardware and tools for model training used in these experiments are as follows:

- YoloV2 and VGG16 - we utilized Java programming using IntelliJ IDE with DL4J beta 6.8 on Ubuntu 18.04 platform with Ryzen 7, 16 GB, with GPU RTX2080Ti, CUDA 10.1, CuDNN7.5. The native model Zoo can be accessed and instantiated directly from DL4J to train YOLOv2 and VGG16 models.
- MobileNetv2 SSD- Google Colaboratory with Python 3.7 and GPU Tesla K80. The Zoo model from TensorFlow was used to train MobileNetv2 with SSD to localize the detected image.

C. Building deep learning models

The model developed in this study involves a fundamental machine learning process and is augmented by the ROI addition phase using DL4J library and TensorFlow frameworks. We chose DL4J due to the computing capability of Python in terms of the programming language and the potential used in actual implementation. This developmental model uses transfer learning techniques [22]. Three models with different platforms were used in this study to compare the model accuracy in recognizing human activity. We focused on the main properties of these models and their set up in DL4J and TensorFlow. DL4J provides a *TransferLearningHelper* class to build VGG16 and YOLOv2 models. The setup steps for VGG16 and YOLOv2 models are illustrated in Fig. 1.

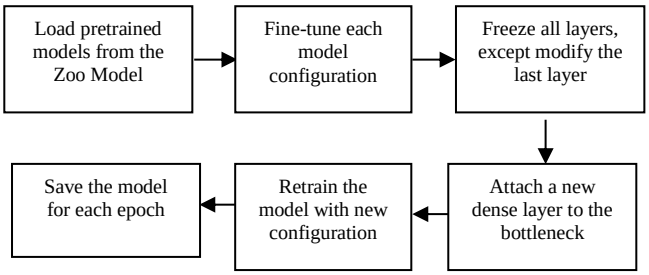


Fig. 1. Steps in the setup of models in DL4J

VGG16: The 16 in VGG16 denotes that the model has 16 layers that have weights. This network has convolution layers of a 3×3 filter with a stride 1. It always uses the same padding and max pool layer with the 2×2 filter of stride two although it has many hyperparameters, which are about 138 million parameters [23].

YOLOv2: In terms of speed, YOLO is one of the best models in object detection and can recognize objects and process frames at a rate up to 150 FPS for small networks. The proposed library is DL4J, and the latest DL4J release provides only YOLOv2. The model uses an architecture as VGG-style, which is Darknet as feature extractors. The proposed paper on YOLOv2 [24] was more accurate and faster than the previous version because it uses some techniques that YOLO did not use, such as batch normalization and anchor boxes. YOLOv2 uses an image input of 416×416 pixels and outputs with 13×13 pixels.

MobileNet V2: This model is suitable for mobile devices because the primary network has a computational cost of 300 million multiply-adds and uses 3.4 million parameters ideal

for any device with low computational power. It has two blocks: residual block with a stride of 1 and block with a stride of 2 for downsizing. Three layers are found for the two types of blocks. If a rectified linear unit is used, then the deep networks only have the power of a linear classifier on the nonzero volume part of the output domain. The model is paired with SSDLite to recognize objects [25]. The model with a new output layer is built by using the TensorFlow framework in Google Colaboratory.

The implementation involves two essential processes, namely, training and recognition. A total of 948 images were used to train the three models. The images were resized in accordance with the input size of each model. A new output layer from the model was added because transfer learning was used, and the other layer was frozen for the model to learn only five new classes. The model parameters were changed on the basis of the dataset used, where the learning rate was set to 0.1 and later reduced by a factor of 0.1 after the saturation of validation loss.

D. Evaluation metrics

The selected seven evaluation metrics used in this study are as follows:

- *Confusion matrix:* A summary of prediction results on a classification problem. It is divided into four parts, which are true positive (TP), true negative (TN), false positive (FP), and false negative (FN).

- *Accuracy:* It determines whether the model is trained correctly and performs properly. The calculation takes all the TP and TN values [26], which is expressed as Eq. 1.

$$Accuracy = (TP+TN) / (TP+FP+FN+TN). \quad (1)$$

- *Precision:* It determines how often the model predicts TPs, which is expressed as Eq. 2. Precision represents a ratio of TPs to the total number of positive predictions [26].

$$Precision = TP / (TP+FP). \quad (2)$$

- *Recall:* Recall helps when the cost of FNs is high, which is expressed as Eq. 3, where the model might ignore the object that is supposed to be detected.

$$Recall = TP / (TP+FN). \quad (3)$$

- *F1 Score* conveys the balance between precision and recall by taking their harmonic mean, which is expressed as Eq. 4.

$$F1= 2 \times (precision \times recall) / (precision+recall). \quad (4)$$

- *Average precision:* a measure that combines recall and precision, which is expressed as Eq. 5, for ranked retrieval results where it is suitable to measure the accuracy of every class for object detectors.

$$AP = \frac{1}{n} \sum_{k=}^{k=n-1} [Recalls(k) - Recalls(k+1)] \times Precision(k),$$

$$Recalls(n) = 0,$$
$$Precision(n) = 1,$$
$$n = Number\ of\ thresholds. \quad (5)$$

- *Mean average precision (mAP):* It obtains the average precision of all classes, which is expressed as Eq. 6 [26].

$$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_k$$

$$APC = Average\ Precision\ of\ class\ k,$$
$$n = Number\ of\ classes. \quad (6)$$

E. Adding ROI phase into the best model

The ROI with a 5×5 matrix size was added to determine the hazard level in the chosen model. Fig. 2 shows the matrix to illustrate the feature map. The grey box represents the danger area because the robot is close to the object, and the black box detects the object. The probabilities of each box are defined on the basis of the object's type and the distance to the robot camera. The further the black box from the grey area, the lower the probability of the area detected. Therefore, an average sum of the area and detected object probabilities were calculated to identify the hazard level, which can be expressed as Eq. 7.

$$Average\ area\ of\ ROI = (\sum_{i=0}^{n} probability\ of\ a\ region)/n$$

$$n = number\ of\ regions. \quad (7)$$

20	40	40	40	20
40	60	60	60	40
60	80	100	80	60
60	80	100	80	60
60	80	100	80	60

Fig. 2. Feature map of the ROI.

IV. Results and Discussion

A. Evaluation Object Detectors on the Test Set

The test sizes of VGG16, YOLOv2, and MobileNetv2 SSD models were set to 0.2 with epochs ranging from 40 to 50. This study used a test set of 380 images to evaluate the model, and comparisons were performed for model evaluation. Model evaluation was performed for each trained model to underfit and overfit data.

Fig. 3 illustrates how the defined classes in the models are detected by using the test set with VGG16. VGG16 shows that most test sets are detected correctly. However, "Walking" has the very least detected class, 0.5, because VGG16 detects "Push Medical Cart" as "Walking" as walking activity occurs while pushing the cart. YOLOv2 shows the least score for all classes compared with VGG16

and MobileNetv2, as illustrated in Fig. 4. Most of the test sets are not detected in YOLOv2. "Standing" is the largest detected class among other classes because its image feature is simple for YOLOv2 to learn and detect. "Standing" in MobileNetv2 has the most significant score, as shown in Fig. 5. However, "Push Medical Cart" has the lowest score, which differs from VGG16 and YOLOv2's least detected class. The confusion matrix diagram shows that more data collection, data augmentation [27], and correct labeling are needed to obtain a promising result.

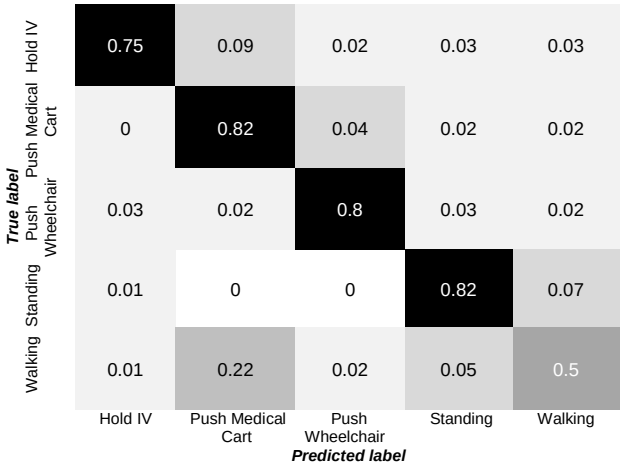


Fig. 3. Confusion matrix of VGG16.

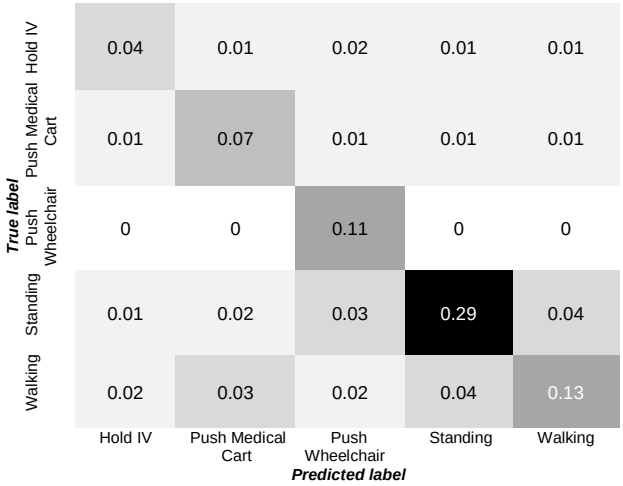


Fig. 4. Confusion matrix of Yolov2.

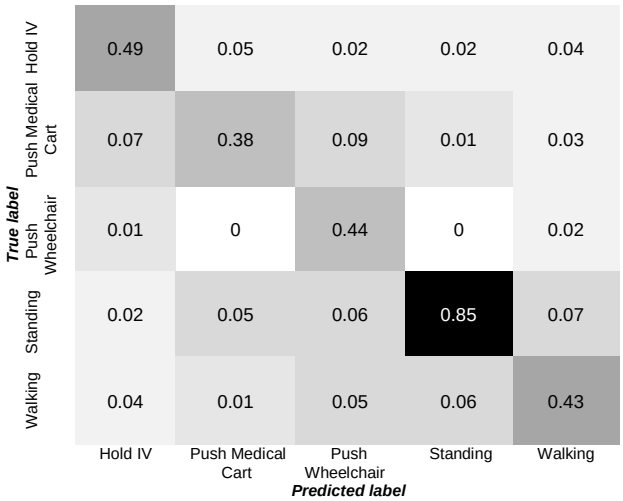


Fig. 5. Confusion matrix of MobileNet v2 SSD.

The evaluation parameters used in the study are precision, recall, and F1-score. The results for each class of the three models are presented in Tables I, II, and III. The evaluation class helper feature in DL4J is used to evaluate the VGG16 model. A customized Python script is developed to evaluate YOLOv2 and MobileNetv2 SSD. Each model has a different score for every class because other models have different architectures for detecting the objects.

TABLE I
PERFORMANCE OF VGG16 MODEL

Class	Precision	Recall	F1-score
0	0.815	0.937	0.872
1	0.911	0.713	0.799
2	0.888	0.909	0.898
3	0.911	0.863	0.886
4	0.625	0.781	0.694

TABLE II
PERFORMANCE OF YOLOV2 MODEL

Class	Precision	Recall	F1-score
0	0.444	0.500	0.470
1	0.636	0.538	0.583
2	1.00	0.578	0.733
3	0.785	0.828	0.805
4	0.541	0.684	0.604

TABLE VI
mAP AND ACCURACY OF OBJEC DETECTOR

Model	*mAP* (%)	Accuracy (%)
VGG16	81.11	93.33
YOLOv2	58.07	46.67
MobileNet v2 SSD	26.67	68.95

The activity in every class might have the same action, such as pushing a wheelchair that requires the human to walk, which leads to misclassification of detection. The conventions of notations used in the class presented in every table are as follows; 0: Hold IV Drip, 1: Push Medical Cart, 2: Push Wheelchair, 3: Standing, and 4: Walking. Table I shows that the "Walking" activity has the lowest score in precision, 0.625, in VGG16. More data are needed because this finding will lead to an overfitted model.

The *mAP* and accuracy for each model are calculated in Table VI. VGG16 had the highest mAP and accuracy compared with YOLOv2 and MobileNet v2 SSD. The accuracies of VGG16, YOLOv2, and MobileNet v2 SSD were 93.33%, 46.67%, and 68.95%, respectively. The YOLOv2 model was chosen in this study because the system requires a bounding box as the location identification of the detected object so that the hazard level can be estimated. The VGG16 model only produces an estimate for each class detected in one image and cannot recognize the detected object's location. VGG16 also uses the sliding windows technique to identify the location of the activity, which is unsuitable if implemented in real-time video.

The MobileNet v2 SSD model was not selected despite having a higher accuracy than YOLOv2 because the *mAP* obtained in the evaluation phase was lower than the YOLOv2 model, which was 26.67%. The model underwent data overfitting because the model only knows the training images rather than the testing images. The platform used for the MobileNetv2 model required high computing costs so that it can run in real-time application.

B. *Evaluation of hazardous level using ROI*

To evaluate the hazardous level produced by the ROI method with a 5×5 matrix, we use 10% of the test set, which is 190 images consisting of all classes using YOLOv2 as an object detector. The low and high hazardous levels are compared with the total number of TP, FP, and FN, as shown in Fig. 6. In this experiment, the ground truth for hazard level for every image is labeled on the basis of the distance between the objects and the camera by manual human estimation.

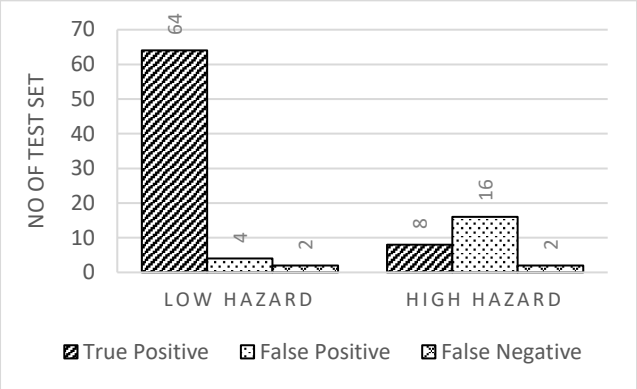


Fig. 6. Evaluation of hazard level with ROI of 5×5 matrix

The graph shows that only 100 images from 10% of the test set are detected with a default confidence threshold of 0.25 and Intersection over Union threshold value of 0.5. The probability from ROI value is only calculated if the model detects the object. If the object is not detected in the test set, then the image is regarded as FN. In Fig. 6, low hazard has the highest value of TP, and FP and FN have the most negligible value. This finding is due to high probabilities defined in the middle of the region, and the positions of standing and walking are mostly in the middle of the test image. Hence, the object is considered a high hazard even if it is far from the camera.

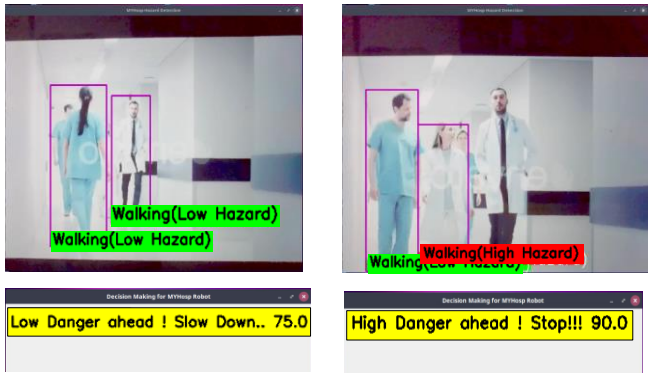
High hazard has more FP than TP where it detects as low rather than high hazard. This finding is because most of the sizes of bounding boxes are large and might cover the whole size of the ROI matrix, including the low region probability. Some images are smaller than the window size due to the image copyright and do not fit the ROI regions. This condition affects the average probability of ROI for an object.

Our goal is to obtain all the TPs for hazardous levels because FP will give a false alarm to the robot and fail to avoid the collision. Thus, high-quality data collection is essential to give the correct hazard level and to produce an accurate object detector model. The ROI size must be varied in accordance with a specific case. The ROI probability defined in this study might be unsuitable for other objects due to the different sizes of the bounding box.

C. *Implementation of hazard level in real-time inference*

The system can detect hazardous human activities for robots in hospitals through real-time video with a resolution of 640×480 pixels. The YOLOv2 model is implemented to detect hazardous human activities in real-time. Each detected activity will be assessed if the hazard level is low or high. Each level will be assigned to the system to decide whether the robot should move slowly or stop immediately, as shown in Figs. 7 and 8. Each activity detected is stored in the user's local folder in the form of pictures and in the format "*Year-Month-Day-Hour- Minute-Second_{DangerLevel}.jpg."* The detected activities are stored for reference to the user. Fig. 8 shows that the *walking and standing* activities classified as "*High Hazard,"* representing human action moving toward the camera. This hazard level made the robot's decision to perform an immediate action to avoid the collision. The model detected "*Standing and Walking"* activities as possible hazards to human activities in real-time and localized the activity by showing a bounding box.

The system's graphical user interface displays the input from a real-time video stream with bounding boxes and classification labels. In Fig. 7, the system detects "*Walking"* and "*Standing"* as the type of human actions displayed from the webcam. Status "*Low Hazard" and "High Hazard"* are detected as the human action is getting far away from the camera, and decision making is shown to give an early warning to the robot.



(a) (b)

Fig. 7. Detection of (a) *low and (b) high hazard* level and decision making for each object.

V. Conclusion

This study focused on the hospital industry to assist hospital officers. Regarding the training model platform, we concluded that the DL4J library has massive potential in training the model robustly although it has limitations in model availability and an evaluation helper class. Some limitations of the system are that it can only recognize objects without occlusion and partial objects in images. The development of this system can be improved with the addition of tracking object IDs for each detected action. Tracking object IDs in images or videos in real-time can make it easier for users to distinguish the activities detected by the system. Overcoming partial object recognition and occlusion is required to improve the system and benefit the system users. An experiment on determining the size of ROI matrices is needed to avoid any false alarms on hazard level and to provide the hazard probability in the region because different bounding box sizes affect the hazard level.

References

[1] S. Wilhelm Pienaar and R. Malekian, "Human Activity Recognition Using Visual Object Detection", arXiv preprint arXiv:1905.03707, 2019.

[2] E.Bulbul, A. Cetin and I. A. Dogru, "Human Activity Recognition Using Smartphones", 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), pp1-6, DOI: 10.1109/ISMSIT.2018.8567275, 2018.

[3] J. P. T. Sien, K. H. Lim and Pek-Ing Au, "Deep Learning in Gait Recognition for Drone Surveillance System". IOP Conference Series: Materials Science and Engineering, vol. 495, pp012031, 2019.

[4] B. A. M.Hossein, S. A. M. Reza, H.Patrick, M. Catherine and A. Farshid, "Robot House Human Activity Recognition Dataset", Proceedings of the 4th UK-RAS Conference: Robotic at Home (#UKRAS21), 2021.
[5] A. Azizi and T. Wong, "Orientation and Scale Based Weights Initialization Scheme for Deep Convolutional Neural Networks", Asia-Pacific Journal of Information Technology & Multimedia, vol. 9, pp103-112, DOI: 10.17576/apjitm-2020-0902-08, 2020.
[6] A. Krizhevsky, I. Sutskever and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", Neural Information Processing Systems, vol. 25, DOI: 10.1145/3065386, 2012.
[7] S. Bhattacharya, V. Shaw, P. K. Singh, R. Sarkar and D.Bhattacharjee, "SV-NET: A Deep Learning Approach to Video Based Human Activity Recognition". Advances in Intelligent Systems and Computing, vol. 1182, 2019.
[8] S. Yang, G. Dejun, L. Fei, M. Luis, D. Houzhu, X. Zhen, H. Randall, K. Adam, C. Shixun, Z. Chengkun and T. Huan, "Robots Under COVID-19 Pandemic: A Comprehensive Survey", IEEE Access, pp1-1, DOI: 10.1109/ACCESS.2020.3045792, 2020.
[9] Y. Guan and W. Mao, "Pedestrian Virtual Space Based Abnormal Behavior Detection". IAENG International Journal of Computer Science, vol. 46, no. 2, pp311-320, 2019.
[10] A. F. M Saifuddin Saif, Z. R. Mahayuddin and H. Arshad, "Vision-Based Efficient Collision Avoidance Model Using Distance Measurement", Soft Computing Approach for Mathematical Modelling of Engineering Problems, pp191-202, DOI: 10.1201/9781003138341- 12-12, 2021.
[11] G. Adam, N. Chris, P. Josh, W. Melanie, B. Alex, K. Vyacheslav, A.Samuel and E. Susan, "Deeplearning4j: Distributed, open-source deep learning for Java and Scala on Hadoop and Spark", DOI: 10.6084/M9.FIGSHARE.3362644.V2, 2016.
[12] F. Al-Azzo, A. M. Taqi and M. Milanova, "Human Related-Health Actions Detection using Android Camera based on TensorFlow Object Detection API", International Journal of Advanced Computer Science and Applications, vol. 9, no. 10, pp9-23, 2018.
[13] E. Cippitelli, F. Fioranelli, E. Gambi, and S. Spinsante, "Radar and RGB-Depth Sensors for Fall Detection: A Review", IEEE Sensors Journal, vol. 17, no. 12, pp3585-3604, DOI: 10.1109/JSEN.2017.2697077, 2017.
[14] Brinjal Suthar and Bijal Gadhia, "Human Activity Recognition Using Deep Learning: A Survey". Data Science and Intelligent Applications, pp217-223, DOI: https://doi.org/10.1007/978-981-15-4474-3_25, 2021
[15] A. Karpathy, G. Toderici, S. Shetty, T.Leung, R. Sukthankar, and Fei- Fei, L, "Large-Scale Video Classification with Convolutional Neural Networks", 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp1725–1732, DOI: 10.1109/CVPR.2014.223, 2014.
[16] M. K. Fiaz and B. Ijaz, "Vision based human activity tracking using artificial neural networks", 2010 International Conference on Intelligent and Advanced Systems, pp1-5, DOI: 10.1109/ICIAS.2010.5716186, 2010.
[17] S. Kahlouche and M. Belhocine, "Human Activity Recognition Based on Ensemble Classifier Model", International Conference on Electrical Engineering and Control Applications, pp1121-1132, 2021.
[18] H. Li, K. Ota, M. Dong and M. Guo, "Learning Human Activities through Wi-Fi Channel State Information with Multiple Access Points", IEEE Communications Magazine, vol. 56, pp124-129, DOI: 10.1109/MCOM.2018.1700083, 2018.
[19] B. Qolomany, A. Al-Fuqaha, D. Benhaddou, and A. Gupta, "Role of Deep LSTM Neural Networks and Wi-Fi Networks in Support of Occupancy Prediction in Smart Buildings", IEEE 19th International Conference on High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), pp50-57, DOI: 10.1109/HPCC-SmartCity-DSS.2017.7, 2017.
[20] D. Albashish, R. Al-Sayyed, A. Abdullah, M. H. Ryalat and N. Ahmad Almansour, "Deep CNN Model based on VGG16 for Breast Cancer Classification", 2021 International Conference on Information Technology (ICIT), pp805-810, DOI: 10.1109/ICIT52682.2021.9491631, 2021.
[21] G. Giuffrida, G. Meoni and L. Fanucci, "A YOLOv2 Convolutional Neural Network-Based Human-Machine Interface for the Control of Assistive Robotic Manipulators", Article Applied Science 2019, vol. 9, no. 11, 2019.
[22] N. A. Rosli, S. N. H. Sheikh Abdullah, A. N. Zamani, A. Ghazvini, N. S. Md Othman and N. A. A. A. Muariff Tajuddin, "Comparison Multi Transfer Learning Models for Deep Fake Image Recognizer", 2021 3rd International Cyber Resilience Conference (CRC), pp1-6, DOI: 10.1109/CRC50527.2021.9392566, 2021.
[23] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", arXiv preprint arXiv:1409.1556v6, 2015
[24] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger". 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp6517-6525, 2017.
[25] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks", 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp4510-4520, DOI: 10.1109/CVPR.2018.00474, 2018.
[26] P. Henderson and V. Ferrari, "End-to-End Training of Object Class Detectors for Mean Average Precision", Asian Conference on Computer Vision, pp198-213, DOI: 10.1007/978-3-319-54193-8_13, 2017.
[27] L. Saima, I. Rosziati, T. Nik, S. Norhalina and S. Suhaila, "Thresholding and Quantization Algorithms for Image Compression Techniques: A Review", Asia-Pacific Journal of Information Technology & Multimedia, vol. 7, pp83-89, DOI: 10.17576/apjitm-2018-0701-07, 2018.