Bulletin of Electrical Engineering and Informatics
Vol. 10, No. 6, December 2021, pp. 3137~3146
ISSN: 2302-9285, DOI: 10.11591/eei.v10i6.2802

Development of 3D convolutional neural network to recognize human activities using moderate computation machine

Malik A. Alsaedi, Abdulrahman S. Mohialdeen, Baraa M. Albaker
College of Engineering, Al-Iraqia University, Sabe' Abkar, Adhamiya, Baghdad, Iraq

Article Info

Article history:

Received Jan 12, 2021
Revised May 20, 2021
Accepted Oct 9, 2021

Keywords:

3D-CNN
Convolutional neural network
Deep learning
HAR
Smart home
Vision

ABSTRACT

Human activity recognition (HAR) is recently used in numerous applications including smart homes to monitor human behavior, automate homes according to human activities, entertainment, falling detection, violence detection, and people care. Vision-based recognition is the most powerful method widely used in HAR systems implementation due to its characteristics in recognizing complex human activities. This paper addresses the design of a 3D convolutional neural network (3D-CNN) model that can be used in smart homes to identify several numbers of activities. The model is trained using KTH dataset that contains activities like (walking, running, jogging, handwaving handclapping, boxing). Despite the challenges of this method due to the effectiveness of the lamination, background variation, and human body variety, the proposed model reached an accuracy of 93.33%. The model was implemented, trained and tested using moderate computation machine and the results show that the proposal was successfully capable to recognize human activities with reasonable computations.

Corresponding Author:

Abdulrahman S. Mohialdeen
College of Engineering
Al-Iraqia University
Sabe' Abkar, Adhamiya, Baghdad, Iraq
Email: abd_saeed@aliraqia.edu.iq

1. INTRODUCTION

HAR is one of the challenging subjects because of the huge number of human activities, some of the activities can be easily noticed some of them are confusing, and some of them require interaction with other objects or humans, besides the diversity of the activities, the recognition methods are also diverse. There are many types of data required to recognize a human activity, some of them use ambient sensors like accelerometer, gyroscope, humidity, and temperature [1], [2]. Some of them get the benefit of the smartphone's sensors like accelerometer and gyroscope [3], [4], the other use the radio frequency [5]. But the most popular recognizing methods use vision-based recognition [6]-[14].

Vision uses images or videos to recognize the activity. Also, there is a lot of challenges for recognizing human activities because of the effect of lamination, variance of background. Still, the question is how to process these visual data to recognize the activity. The answer there are many techniques most of them use machine learning, and deep learning has shown an excellent benefit for recognizing human activities, especially the CNN which are very useful for vision-based data recognition.

In this paper, we will design a neural network architect a.k.a. model, that can be used for human activity monitoring, the model proposed of 3D dimensional CNN (3D-CNN), and the purpose of using 3D-CNN is to extract spatial and temporal features rather than only spatial features, and the activity consists

of multiple movements that can be identified by extracting the temporal features from between several numbers of frames.

Our proposed model is a small number of 3D-CNN layers to reduce the amount of processing time so that any computer with low computational ability could recognize human activity online without delay. CNN was introduced by Fukushima [12]. At the beginning, it was mainly used for image classification, and image features extraction, one of the most attractive models that been invented to get into a challenge to classify images [13], [14]. During that many video human activity datasets were published and KTH [15] is one of the most popular small size datasets. CNN architect encouraged researchers to use it in HAR with image data taken from video dataset [10], [16]. Other researchers proposed to use two-stream CNN where the first stream is an image data and the second it's optical flow [16], [17] 3D-CNN for HAR most cited articles [18] proposed model with TRECVID dataset, and [19] proposed 3D-CNN model for UCF-101 dataset [20] in which our model inspired by their model.

2. RESEARCH METHOD

This paper proposing to use deep neural networks (DNN) models that consists of many different layers, and each layer has its purpose during training and testing. Still, the dominant layer in which the article focused on is the Convolutional neural network, which is one of the most widely used neural networks, and its central idea is applying filters to the input data or convolute it, and transfer the convoluted data to the next layer. CNN was primarily used to deal with image data. Now there are 1D, 2D, and 3D CNN to get the benefit of this architect for another type of data with a different number of dimensions. 3D-CNN used for three-dimensional data which is very suited for our project, because we are dealing with video data, and the reason for using video data rather than image data is the activity made of several consequential movements of body parts. This continuous movement can be noticed with successive images, and this is a video.

Pooling layers used in this paper are max-pooling, which returns the maximum value within a kernel size when it wraps around the data, average-pooling which returns the average amount within a kernel size that wraps along with the data, and global-average-pooling, which returns the average value of each dimension or each kernel in the CNN layer, and that is why it is useful at the final part of the model to reduce the number of parameters, and to help the model overcome overfitting dropout layers are used.

2.1. Proposed models

The first suggested model shown in Figure 1 is influenced by the model proposed by Tran *et al.* [19]. The model consists of three connected 3D convolutional neural networks with 3*3*3 kernel size for all convolutional layers, 6ed by a max-pooling layer with a kernel size of 2*2*2 for all max-pooling layers. Zero-padding is used for all convolutional layers, and 'ReLU' activation function added after each convolutional and fully connected (FC) layers except for the last was SoftMax.

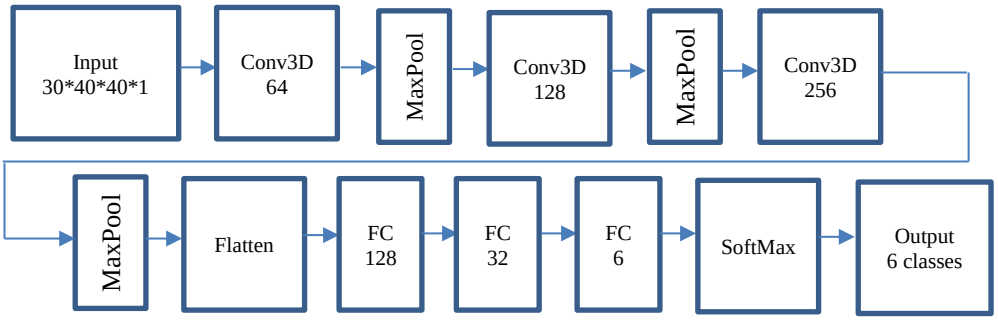


Figure 1. Initial proposed model

In the second attempt, other convolutional and max-pooling layers added before flatten layer, to increase the accuracy. Then dropouts were added in different places in the model with five attempts to get high accuracy, where all dropouts were with a 0.5 percentage factor. We try with our model to reduce the number of parameters (weights and biases) and return to the first model with dropouts. The accuracy increased using the dropouts. But, the number of parameters in huge, because the output of the layer before flatten layer was more extensive than before removing convolutional and pooling layers, and as the pooling layer is gone the number of parameters increased, in (1)-(3) shows how the number of parameters calculated.

So, to solve this problem GlobalAveragePooling3D layer replaced the flatten layer, this replacement reduced the number of parameters to about two million parameters, which is very helpful for low or moderate computation capabilities machines to deal with online activity recognition.

No. of parameters in CNN=($filter_{width}$ * $filter_{height}$ * $filter_{depth}$ + 1) * no. of filters (1)

No. of parameters in FC net=neurons of current layer * neurons of previous layer (2)

No. of parameters in flatten=multiplication of all previous layer dimension (3)

3. RESULTS AND DISCUSSION

The neural network model is trained on the KTH dataset, the dataset is doubled before using it by adding a flipped copy of it, 20% of data is taken for test after training, and another 20% taken for the test during the training, and 60% were taken for the training process. The model was trained using Tensorflow-v2.1 [21] as backend, and Keras-v2.3.1 [22] using Python-language-v3.7.5 as front-end, with batch size=16 and the shape of the frame taken from the video were 40x40 pixels with one channel (grayscale), from each video 30 frames were taken between each frame and another there were four frames between taken frames discarded.

The optimizer of the model was Adam optimizer [23], and with a learning rate of 0.001 and categorical cross-entropy loss function, the machine specification was: HP 15 Notebook, Memory: 12288MB RAM, Intel-Core i5-3230M processor with four cores, two of them are physical, and the maximum frequency is 2.6GHz and Windows 8.1 Enterprise 64-bit OS (6.3, build 9600).

Validation data controls the training operation. So, if an update made to the model and validation data applied to the model and the losses did not improved for three epochs for the learning rate would be multiplied by a half and the minimum reduction is 0.0001. If the losses did not improve for 15 epochs consequently, the training would be finished before getting to the given number of epochs is 100.

3.1. Calculating results

After training operation finishes test samples are pushed to the model to get the response the accuracy, precision, recall, and f1_score are calculated using (4)-(7) which uses confusion matrix shown in Figure 2 [24], for average loss is calculated using categorical cross-entropy algorithm [25].

Data class	Classified as *pos*	Classified as *neg*
pos	true positive (*tp*)	false negative (*fn*)
neg	false positive (*fp*)	true negative (*tn*)

$$\begin{bmatrix} tp & fn \\ fp & tn \end{bmatrix}$$

Figure 2. Confusion matrix annotation

$$\text{Accuracy}=\frac{\sum_{i=1}^{l} tp_i+tn_i}{\sum_{i=1}^{l}(tp_i+fn_i+fp_{i+tn_i})} \quad (4)$$

In (5) shows the way to calculate accuracy which defines the effectiveness of the model overall.

$$\text{Precision}=\frac{\sum_{i=1}^{l} tp_i}{\sum_{i=1}^{l}(tp_i+fp_i)} \quad (5)$$

According to (5) shows the way to calculate precision which determines the matching between the label of classes and the calculated labels. In (6) shows the way to calculate recall which demonstrates the effectiveness of the model to identify the label of classes. As shown in (7) shows the way to calculate F1_score which defines the relation between output data taken from the model after entering data for test and the positive labels.

In (8) show the way to calculate the average loss, where N is the number of samples, M is the number of classes, d is the true label or desired output, and y is the calculated or tested output from the model. Table 1 shows the calculated results, and it's figure number. Table 2 shows a comparison of accuracies for several studies done on the KTH dataset for human activity recognition and our study accuracy, and it is evident that our method has shown a remarkable improvement according to accuracy.

$$\text{Recall}=\frac{\sum_{i=1}^{l} tp_i}{\sum_{i=1}^{l}(tp_i+fn_i)} \quad (6)$$

$$\text{F1\_score}=\frac{(\beta^2+1)*\text{Precision}+\text{Recall}}{\beta^2*\text{Precision}+\text{Recall}} \quad (8)$$

$$\text{Loss}=\frac{\sum_{i=1}^{N}-\frac{1}{M}\sum_{j=1}^{M} d_{i,j}\log(y_i,j)}{N} \quad (9)$$

Table 1. Calculated results for all models

No.	No. of figure	Accuracy %	Loss	Precision %	Recall %	F1_score %
1.	Figure 3	85.83	0.39	86.30	85.89	86.09
2.	Figure 4	88.75	0.39	88.97	88.97	88.97
3.	Figure 5	92.08	0.23	92.67	91.90	92.28
4.	Figure 6	92.92	0.22	93.08	92.95	93.02

Table 2. Comparison of accuracies of researches done on KTH

No.	Method	Accuracy %
1	Ahmad and Lee [26]	84.83
2	Taylor *et al*. [27]	88.00
3	Qian *et al*. [28]	88.69
4	Our method	93.33

Number the table consecutively according to the first mention (sequential order).

3.2. Calculating the number of operations for layers

We want to calculate an approximate number of operations for each layer, the calculations don't include controlling operations, calculations are detailed below:

a. 3D-CNN each kernel convolutes on the entire input data, and for 3D-CNN with a kernel size of (Kd, Kh, Kw), input data of (frames, height, width) and strides are (strided, strideh, stridew), we would have:

No. of operations per node=(Kd * Kh * Kw + 1)2 * no. of previous kernels (9)

Output nodes=((frames-Kd)/strided+1) * (height-Kh)/strideh+1) *((width-Kw)/stridew+1) *no. of current kernels (10)

Output nodes=(frames/strided+1) * (height/strideh+1) * (width/stridew+1) * no. of current kernels (11)

In (9) shows the number of operations for each output node came from the convolution operation and the power two because we have multiplications, in (10) [29] shows the number of output node and for each output node we have. For 3D-CNN layer, but for no zero paddings, if we use padding which is used in our proposed model the number of operation would be as shown in (11) [29].

b. Maxpooling3D performs comparison operation, for a pool window size of (Pd, Ph, Pw), input data of (frames, height, width) and strides are (strided, strideh, stridew), we would have:

No. of operations per node=Pd * Ph * Pw (12)

Output nodes=((frames-Pd)/strided+1) * ((height-Ph)/strideh+1) * ((width-Pw)/stridew+1) no. of current kernels (13)

Output nodes=(frames/strided+1) * (height/strideh+1) * (width/stridew+1) * current kernels (14)

In (12) shows the number of operations for each pool window, in (13) shows the number of output node without padding which is used in our model and the size of the stride are the same pool window size, in (14) shows the number of output nodes if there are zero paddings.

c. Fully connected has a vast number of parameters as compared with CNN if the number of neurons of the previous layer is Nprevious and the number of neurons of the current layer is Ncurrent.

In (15) shows the number of operations for a fully connected layer, power two because we have multiplications.

Number of operations=(Nprevious * Ncurrent)2 (15)

d. Flatten has only one operation which is reshaping the dimensions into one dimension.
e. Dropout works only during training operation, and it's just hidden randomly chosen a portion of nodes so as not to participate in producing the output at only some point in the training, so for testing it doesn't cost any operations.
f. GlobalAveragePooling3D adds nodes for each channel where the previous layer output is (frames, height, width, channels), so it adds all the numbers in frames, height, and width for a particular channel and divides by the number of (frames * height * width), so the total number of operations is shown in (16).

Number of operations=(frames * height * width) * channels (16)

Table 3 shows the number of operations for each model proposed according to their figures, and we can see the least number of operations is the model with the least number of parameters and high accuracy of 92.92%.

Table 3. Number of operations for each model according to its figure

No. of figure	No. of operations
Figures 3	$1.08 * 10^{13}$
Figures 4, 5	$5.77 * 10^{12}$
Figure 6	$4.77 * 10^{12}$

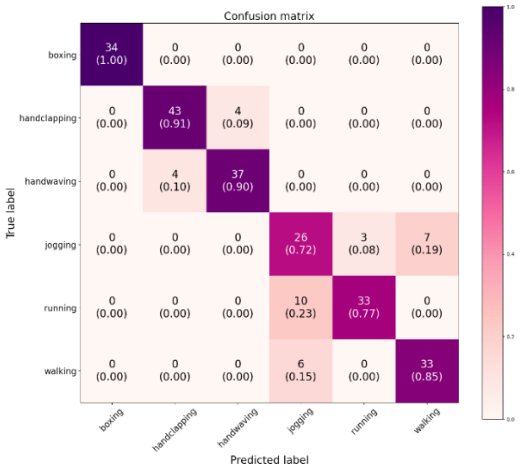
4. DISCUSSION

Each result is discussed according to the number of figures.

Figure 4 shows model architect, confusion matrix, accuracy, and loss figures for the earlier proposed model. We can notice that because the last before flatten were not small we got a massive number of the parameters for the fully connected layer, also we can see that the model's learning was saturated in early time which completed learning within 30 epochs. Figure 5 shows model architect, confusion matrix, accuracy and loss figures for the proposed model after adding Conv3D and MaxPooling before flatten to reduce the number of parameters, and the training time and the training finished within 30 epochs. Figure 6, shows model architect, confusion matrix, accuracy, and loss figures for the model after adding dropouts after the fourth, sixth, and eighth layers, the accuracy for this model shown remarkable improvement and the training finished in epoch 100 which is the final demanded epoch. Figure 6 shows model architect, confusion matrix, accuracy, and loss figures for the model after changing flatten layer with GlobalAveragePooling3D, which reduced the number of parameters and so the training time per epoch. It also reduced the number of operations during testing and got a fantastic accuracy of 92.92%, the training was finished in epoch 82.

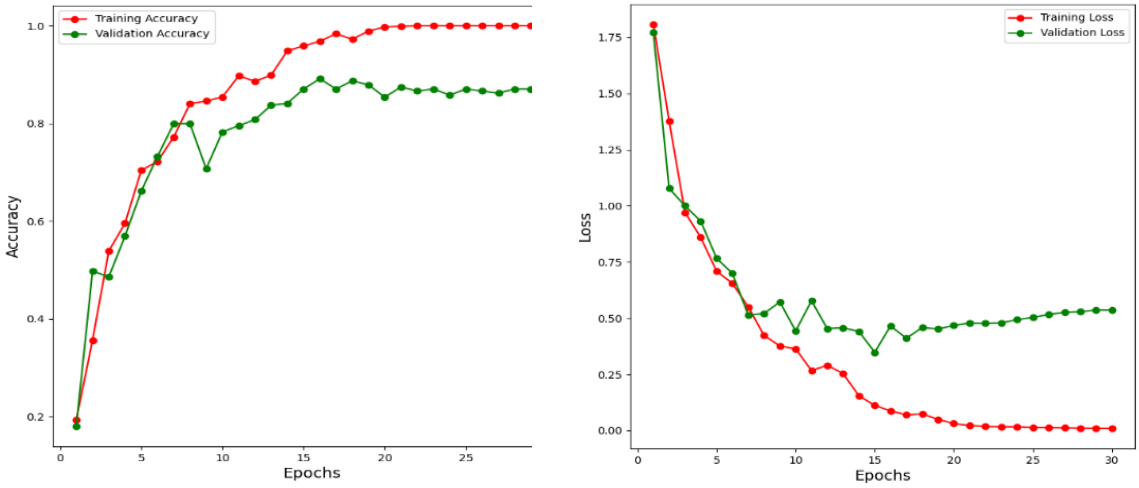
Layer (type)	Output shape	Parameters
Conv3D	30, 40, 40, 64	1792
MaxPooling3D	15, 20, 20, 64	0
Conv3D	15, 20, 20, 128	221312
MaxPooling3D	7, 10, 10, 128	0
Conv3D	7, 10, 10, 256	884992
MaxPooling3D	3, 5, 5, 256	0
Flatten	19200	19200
Dense	128	2457728
Dense	32	4128
Dense	6	198
Total Parameters		3,570,150

(a)



(b)

Figure 3. Earlier proposed model; (a) model architecture, (b) confusion matrix

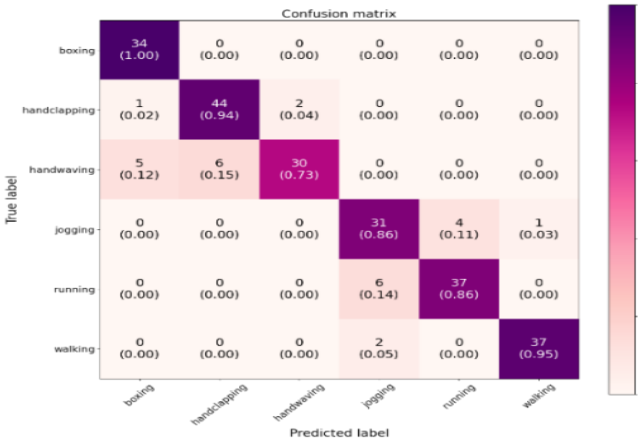


(c) (d)

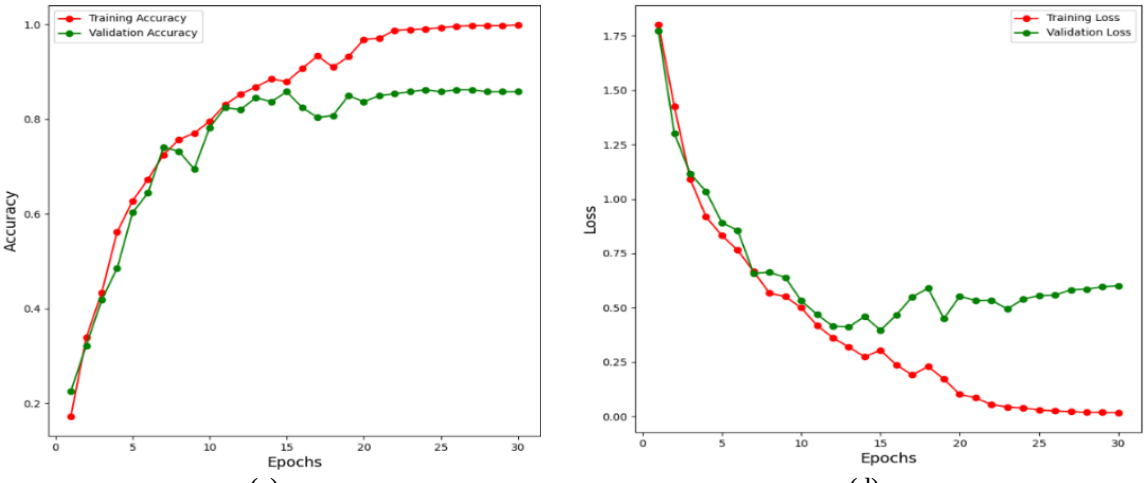
Figure 4. Earlier proposed model; (c) accuracy of training and validation, (d) losses of training and validation (*continue*)

Layer (type)	Output shape	Parameters
Conv3D	30, 40, 40, 64	1792
MaxPooling3D	15, 20, 20, 64	0
Conv3D	15, 20, 20, 128	221312
MaxPooling3D	7, 10, 10, 128	0
Conv3D	7, 10, 10, 256	884992
MaxPooling3D	3, 5, 5, 256	0
Conv3D	3, 5, 5, 256	1769728
MaxPooling3D	1, 2, 2, 256	0
Flatten	1024	0
Dense	128	131200
Dense	32	4128
Dense	6	198
Total Parameters		3,013,350

(a)



(b)

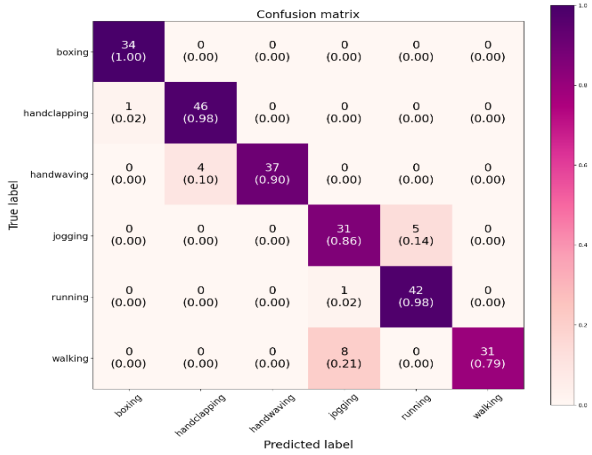


(c) (d)

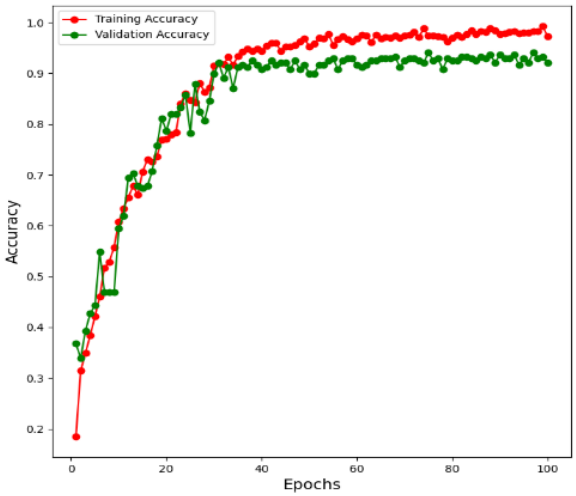
Figure 5. Adding fourth convolutional and max-pooling layers; (a) model architecture, (b) confusion matrix, (c) accuracy of training and validation, (d) losses of training and validation

Layer (type)	Output shape	Parameters
Conv3D	30, 40, 40, 64	1792
MaxPooling3D	15, 20, 20, 64	0
Conv3D	15, 20, 20, 128	221312
MaxPooling3D	7, 10, 10, 128	0
Dropout (0.5)	7, 10, 10, 128	0
Conv3D	7, 10, 10, 256	884992
MaxPooling3D	3, 5, 5, 256	0
Dropout (0.5)	3, 5, 5, 256	0
Conv3D	3, 5, 5, 256	1769728
MaxPooling3D	1, 2, 2, 256	0
Dropout (0.5)	1, 2, 2, 256	0
Flatten	1024	0
Dense	128	131200
Dense	32	4128
Dense	6	198
Total Parameters		3,013,350

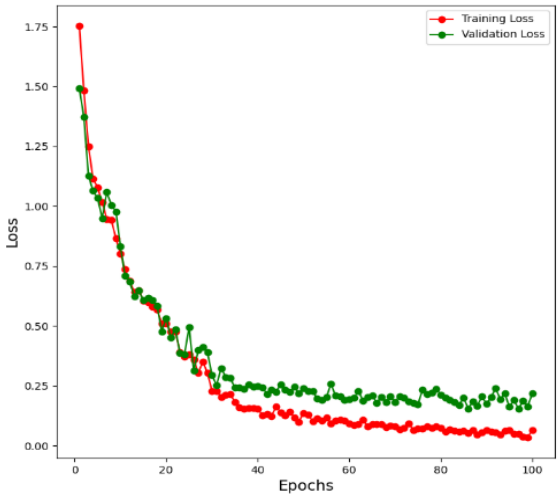
(a)



(b)



(c)

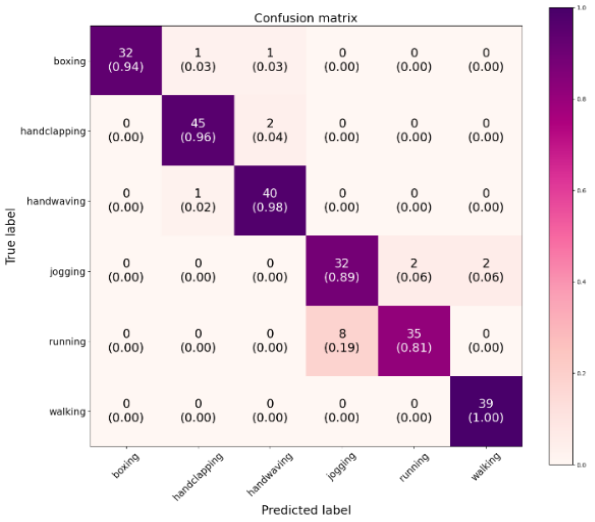


(d)

Figure 6. Dropouts before third and fourth convolutional and also before flatten layer; (a) model architecture, (b) confusion matrix, (c) accuracy of training and validation, (d) losses of training and validation

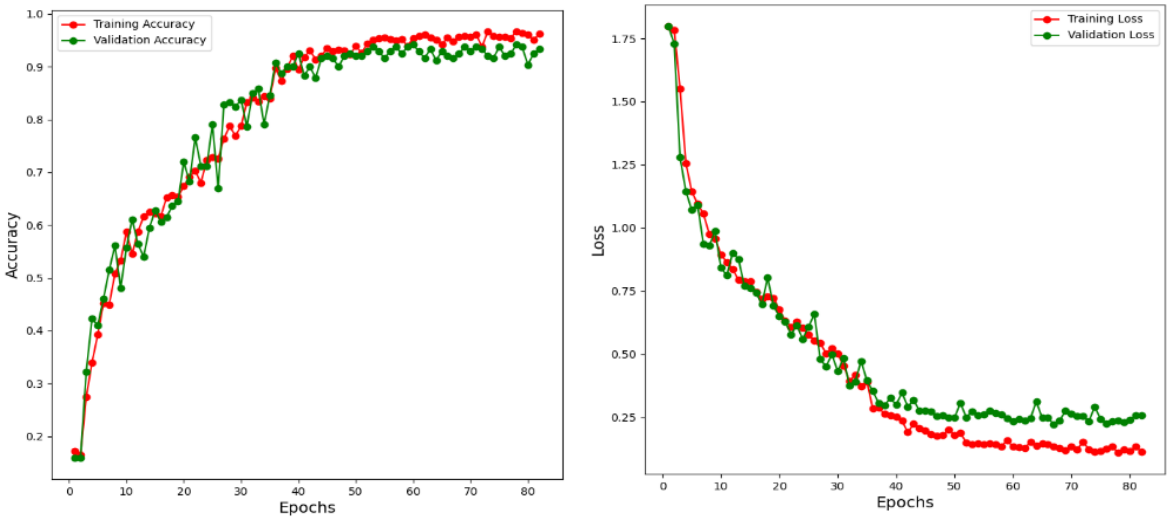
Layer (type)	Output shape	Parameters
Conv3D	30, 40, 40, 64	1792
MaxPooling3	15, 20, 20, 64	0
Conv3D	15, 20, 20, 128	221312
MaxPooling3	7, 10, 10, 128	0
Conv3D	7, 10, 10, 256	884992
MaxPooling3	3, 5, 5, 256	0
Dropout (0.5)	3, 5, 5, 256	0
GlobalAVG3D	256	0
Dropout (0.5)	256	0
Dense	128	32896
Dense	32	4128
Dense	6	198
Total parameters		1,145,318

(a)



(b)

Figure 6. Replacing flatten with global average pooling; (a) model architecture, (b) confusion matrix



(c) (d)

Figure 6. Replacing flatten with global average pooling; (c) accuracy of training and validation, (d) losses of training and validation (*continue*)

We can see that dropout has great benefit, but this benefit can't be taken unless when we put dropout in the right place, we can see that there were several changes to the place and number of dropouts when had seen that dropouts increases the accuracy when it was added before the third convolutional and flatten layers but decreased slightly decreased when added before fourth convolutional layer. Then the place of dropouts was changed to be before and after flatten layer, in which we got the maximum accuracy, after that this increasing tested for the model with a smaller number of layers for the aim of decreasing the number of parameters. The results were helpful, then complete the operation of parameters decreasing, flatten layer has been replaced by Global-average-pooling, which reduced the number of parameters for the model by two million parameters.

5. CONCLUSION

We have designed a model that can be used for online human activity recognition using moderate computation machine. The accuracy of our proposed model was raised to 93.33%, and 92.92% for the model with reduced amount of parameters. The last presented model is useful for moderate computation capabilities machines, due to its low number of parameters and a low number of mathematical operations. We have reached this high accuracy by getting the benefit of dropouts, and decreasing learning rate during training when there is no improvement. The model with a low number of mathematical operations could be used for online human activity recognition in a smart houses, helping monitoring human activities in the houses. We intend to do more augmentation for the data to increase the overall accuracy, where only flipping augmentation is made to the data.

REFERENCES

[1] X. Zhou, W. Liang, K. I. Wang, H. Wang, L. T. Yang and Q. Jin, "Deep-Learning-Enhanced Human Activity Recognition for Internet of Healthcare Things," in *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6429-6438, July 2020, doi: 10.1109/JIOT.2020.2985082..

[2] V. Bianchi, M. Bassoli, G. Lombardo, P. Fornacciari, M. Mordonini and I. De Munari, "IoT Wearable Sensor and Deep Learning: An Integrated Approach for Personalized Human Activity Recognition in a Smart Home Environment," in *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8553-8562, Oct. 2019, doi: 10.1109/JIOT.2019.2920283.

[3] A. K. M. Masum, A. Barua, E. H. Bahadur, M. R. Alam, M. A. U. Z. Chowdhury and M. S. Alam, "Human Activity Recognition Using Multiple Smartphone Sensors," *2018 International Conference on Innovations in Science, Engineering and Technology (ICISET)*, 2018, pp. 468-473, doi: 10.1109/ICISET.2018.8745628.

[4] M. M. Hassan, M. Z. Uddin, A. Mohamed and A. Almogren, "A robust human activity recognition system using smartphone sensors and deep learning," *Future Generation Computer Systems*, vol. 81, pp. 307-313, 2018, doi: 10.1016/j.future.2017.11.029.

[5] X. Wu, Z. Chu, P. Yang, C. Xiang, X. Zheng and W. Huang, "TW-See: Human Activity Recognition Through the Wall With Commodity Wi-Fi Devices," in *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 306-319, Jan. 2019, doi: 10.1109/TVT.2018.2878754.
[6] A. Diba *et al.*, "Temporal 3D ConvNets: New Architecture and Transfer Learning for Video Classification," *Computer Science*, 2017.
[7] T. Lima, B. Fernandes and P. Barros, "Human action recognition with 3D convolutional neural network," 2017 *IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, 2017, pp. 1-6, doi: 10.1109/LA-CCI.2017.8285700.
[8] J. Carreira and A. Zisserman, "Quo Vadis, action recognition? A new model and the kinetics dataset," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2017-Janua, pp. 4724–4733, 2017.
[9] R. Singh, A. K. S. Kushwaha and R. Srivastava, "Multi-view recognition system for human activity based on multiple features for video surveillance system," *Multimedia Tools and Applications*, vol. 78, no. 12, pp. 17165-17196, 2019, doi: 10.1007/s11042-018-7108-9.
[10] H. D. Mehr and H. Polat, "Human Activity Recognition in Smart Home With Deep Learning Approach," *2019 7th International Istanbul Smart Grids and Cities Congress and Fair (ICSG)*, 2019, pp. 149-153, doi: 10.1109/SGCF.2019.8782290.
[11] Z. Tu *et al.*, "Multi-stream CNN: Learning representations based on human-related regions for action recognition," *Pattern Recognition*, vol. 79, pp. 32-43, 2018, doi: 10.1016/j.patcog.2018.01.020.
[12] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193-202, 1980, doi: 10.1007/BF00344251.
[13] J J. Deng, W. Dong, R. Socher, L. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.
[14] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 60, no. 6, pp. 84-90, 2017, doi: 10.1145/3065386.
[15] "KTH dataset," 2005. [Online]. Available: https://www.csc.kth.se/cvap/actions/. [Accessed: 27-May-2020].
[16] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar and L. Fei-Fei, "Large-Scale Video Classification with Convolutional Neural Networks," *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725-1732, doi: 10.1109/CVPR.2014.223.
[17] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proceedings of the 27th International Conference on Neural Information Processing Systems*, vol. 1, no. 1, pp. 568-576, 2014, doi: 10.5555/2968826.2968890.
[18] S. Ji, W. Xu, M. Yang and K. Yu, "3D Convolutional Neural Networks for Human Action Recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221-231, Jan. 2013, doi: 10.1109/TPAMI.2012.59.
[19] D. Tran, L. Bourdev, R. Fergus, L. Torresani and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, vol. 2015 Inter, pp. 4489-4497, 2015.
[20] K. Soomro, A. R. Zamir and M. Shah, "UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild," *Computer Vision and Pattern Recognition*, no. November, 2012.
[21] M. Abadi *et al.*, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," *Distributed, Parallel, and Cluster Computing*, 2016.
[22] François Chollet, "Keras," 2015. [Online]. Available: https://keras.io/. [Accessed: 08-Jun-2020].
[23] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," *Computer Science, Mathematics*, pp. 1-15, 2015.
[24] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427-437, 2009, doi: 10.1016/j.ipm.2009.03.002.
[25] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," *32nd Conference on Neural Information Processing Systems (NeurIPS)*, 2018, vol. 2018-Decem, no. NeurIPS, pp. 8778-8788, doi: 10.5555/3327546.3327555.
[26] M. Ahmad and S. W. Lee, "Human action recognition using shape and CLG-motion flow from multi-view image sequences," *Pattern Recognition*, vol. 41, no. 7, pp. 2237-2252, 2008, doi: 10.1016/j.patcog.2007.12.008.
[27] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler, "Convolutional learning of spatio-temporal features," *European conference on computer vision*, Springer, Berlin, Heidelberg, 2010, vol. 6316 LNCS, no. PART 6, pp. 140-153, doi: 10.1007/978-3-642-15567-3_11.
[28] H. Qian, Y. Mao, W. Xiang and Z. Wang, "Recognition of human activities using SVM multi-class classifier," *Pattern Recognition Letters*, vol. 31, no. 2, pp. 100-111, 2010, doi: 10.1016/j.patrec.2009.09.019.
[29] I. Vasilev, D. Slater, G. Spacagna, P. Roelants and V. Zocca, "Python Deep Learning," 2nd Editio. Birmingham: Packt Publishing, 2019.

BIOGRAPHIES OF AUTHORS

Malik Alsaedi is Asst. Prof. of electrical engineering. He finished his B.Sc. degree from University of Technology University \ Baghdad, the M.Tch. degree from JNTU University\ India and the Ph.D. degree from UTM university \ Malaysia. Currently position a deputy dean of engineering college \ Al-Iraqia University\ Iraq. He is interested in optical communication and IoT technology.

Abdulrahman S. Mohialdeen has a Bachelor degree in Electrical Engineering from University of Baghdad, Master degree in Computer Engineering from Al-Iraqia University, research interest in deep learning, human activity recognition, and computer vision.

Baraa Munqith Albaker received both B.Sc. degree in electrical engineering and M.Sc. degree in computer and control engineering from University of Baghdad, Iraq, and Ph.D. degree in control engineering from University of Malaya, Malaysia. He had worked in industry on data acquisition systems and radar signal processing and analysis for over three years. He was a lecturer at University of Baghdad for four years. Next, he was a senior lecturer of UMPEDAC research Centre, University of Malaya for two years. Currently, he works as head of Networks Engineering department at Al-Iraqia University. His research interests focus on contemporary development in computer and control applications.