Cluster Computing
https://doi.org/10.1007/s10586-021-03348-7

New CNN and hybrid CNN-LSTM models for learning object manipulation of humanoid robots from demonstration

Simge Nur Aslan[1] · Recep Özalp[1] · Ayşegül Uçar[1] · Cüneyt Güzeliş[2]

Received: 15 March 2021 / Revised: 5 June 2021 / Accepted: 19 June 2021

Abstract
As the environments that human live are complex and uncontrolled, the object manipulation with humanoid robots is regarded as one of the most challenging tasks. Learning a manipulation skill from human Demonstration (LfD) is one of the popular methods in the artificial intelligence and robotics community. This paper introduces a deep learning based teleoperation system for humanoid robots that imitate the human operator's object manipulation behavior. One of the fundamental problems in LfD is to approximate the robot trajectories obtained by means of human demonstrations with high accuracy. The work introduces novel models based on Convolutional Neural Networks (CNNs), CNNs-Long Short-Term Memory (LSTM) models combining the CNN LSTM models, and their scaled variants for object manipulation with humanoid robots by using LfD. In the proposed LfD system, six models are employed to estimate the shoulder roll position of the humanoid robot. The data are first collected in terms of teleoperation of a real Robotis-Op3 humanoid robot and the models are trained. The trajectory estimation is then carried out by the trained CNNs and CNN-LSTM models on the humanoid robot in an autonomous way. All trajectories relating the joint positions are finally generated by the model outputs. The results relating to the six models are compared to each other and the real ones in terms of the training and validation loss, the parameter number, and the training and testing time. Extensive experimental results show that the proposed CNN models are well learned the joint positions and especially the hybrid CNN-LSTM models in the proposed teleoperation system exhibit a more accuracy and stable results.

Keywords Humanoid robots · Learning from demonstration · Convolution neural networks · Long short-term memory network · Object manipulation

1 Introduction

Learning from Demonstration (LfD) is a popular research method in which the robots automatically learn the trajectories to be followed from human demonstrations [1–3]. The method is suitable for complex manipulation tasks. It facilitates to encode and transfer the knowledge from the human experts to the robot even in uncontrolled environments such as houses and hotels, and hospitals that are outside of factories [1–3]. LfD should not require the kinematics model of the robot and previous programming of the joint motors [4].

There are two basic kinds of LfD as the kinesthetic guidance and teleoperation [5]. The kinesthetic guidance is separated into two groups [6, 7]. In the first one, the human operator manually moves the robot joints to realize the desired task and the obtained joint positions are saved and modelled by artificial intelligence learning methods for being imitated by the robot in the feature [8–14]. A drawback of the method is to move all joints by a high precision at the same time. In the second kinesthetic method, the human operator wears the external motion sensors or moves his arms and legs or does the same

✉ Ayşegül Uçar
agulucar@firat.edu.tr

Simge Nur Aslan
simgeaslan124@gmail.com

Recep Özalp
rozalp@firat.edu.tr

Cüneyt Güzeliş
cuneyt.guzelis@yasar.edu.tr

[1] Mechatronics Engineering Department, Firat University, 23119 Elazig, Turkey

[2] Electrical and Engineering Department, Yaşar University, 35100 Izmir, Turkey

Published online: 26 June 2021

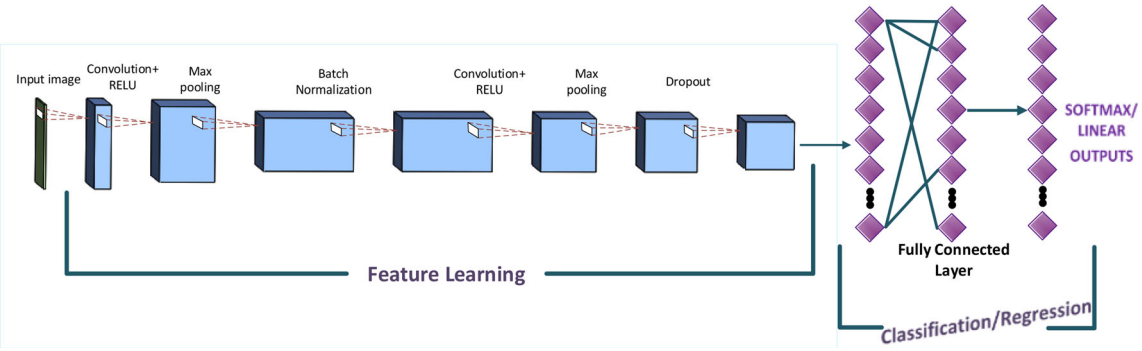


Fig. 1 Schematic flow diagram of convolutional neural networks

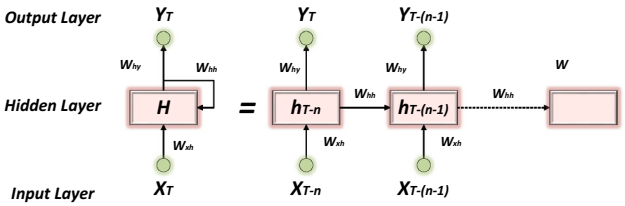


Fig. 2 A conventional recurrent neural network structure [40]

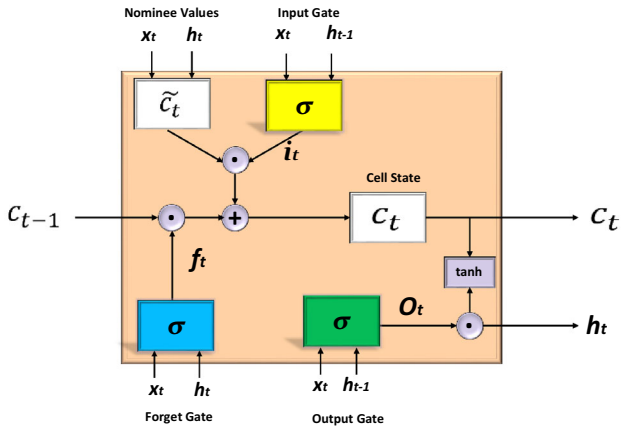


Fig. 3 LSTM memory cell [40, 41]

movements in front of the visual sensors such as the Kinect camera [15]. The disadvantages of the method are that an additional skeleton following sensor should be used and the transformation to the robot from the sensor coordinate system is not always possible. For example, the Pepper and Nao humanoid robots have different leg and hand joint structures from humans, respectively.

In teleoperation, the robot is controlled by a human from a remote distance by using the peripherals such as joysticks, keyboard, and cellphone [16]. In [17], the teleoperation and kinematics guidance on Pepper robot are comparatively presented by constructing virtual reality. A solution using a brain-computer interface system is

Table 1 The structure of the Model 1

Type (CNN)	Output	Filter-Stride	Feature map
Input	360,640	–	3
Conv_1	178,178	5 × 5,2	12
Conv_2	87,157	5 × 5,2	24
MaxPooling_1	43,78	2 × 2,1	24
Conv_3	20,37	5 × 5,2	36
Conv_4	8,17	5 × 5,2	48
MaxPooling_2	4,8	2 × 2,1	48
Dropout	4,8	–	48
Flatten_1	1	–	1536
Dense_1	1	–	100
Dense_2	1	–	50
Dense_3	1	–	30
Dense_4	1	–	1

Table 2 The structure of the Model 2

Type (CNN) (3 × 1)(1 × 3)	Output	Filter-Stride	Feature map
Input	360,640	–	3
Conv_1	179,320	5 × 5,2	12
Conv_2	90,159	3 × 3,2	24
MaxPooling_1	45,79	2 × 2,1	24
Conv_3	22,40	5 × 5,2	36
Conv_4	11,19	3 × 3,2	48
MaxPooling_2	5,9	2 × 2,1	48
Dropout	5,9	–	48
Flatten_1	1	–	2160
Dense_1	1	–	100
Dense_2	1	–	50
Dense_3	1	–	30
Dense_4	1	–	1

Table 3 The structure of the Model 3

Type (CNN) (5 × 5)(3 × 3)	Output	Filter-Stride	Feature map
Input	360,640	–	3
Conv_1	178,318	3 × 1,2	12
Conv_2	88,158	1 × 3,2	24
MaxPooling_1	44,79	2 × 2,1	24
Conv_3	20,38	3 × 1,2	36
Conv_4	9,18	1 × 3,2	48
MaxPooling_2	4,9	2 × 2,1	48
Dropout	4,9	–	48
Flatten_1	1	–	1728
Dense_1	1	–	100
Dense_2	1	–	50
Dense_3	1	–	30
Dense_4	1	–	1

Table 4 The structure of the Model 4

Type (CNN + LSTM)	Output	Filter-Stride	Feature map
Input	360,640	–	3
Conv_1	178,178	3 × 1,2	12
Conv_2	87,157	1 × 3,2	24
MaxPooling_1	43,78	2 × 2,1	24
Conv_3	20,37	3 × 1,2	36
Conv_4	8,17	1 × 3,2	48
MaxPooling_2	4,8	2 × 2,1	48
Dropout	4,8	–	48
TimeDistributed_Flatten_1	4	–	384
LSTM	1	–	36
Dense_1	1	–	100
Dense_2	1	–	50
Dense_3	1	–	30
Dense_4	1	–	1

proposed for a biofeedback and a humanoid robot for guidance effectively throughout remote control in [18].

The teleoperation system should not require an external sensor and a big experimental setup. Hence, this paper uses the teleoperation method of LfD for object manipulation with the humanoid robots. Most important challenge in LfD is to approximate the trajectories obtained from human demonstrations with high accuracy and fast in real time [19–21]. In [22], a coactive learning framework using iterative steps was proposed. [23] Introduced a framework based on the targeted feature queries. [19, 24] Introduced human-grounded concepts by fusing predicates associated with segments of a trajectory.

This paper proposes a solution based on deep learning. Deep learning models approximate better data and so provide higher estimation performance than the conventional machine learning methods in the testing stage [25]. Recently, a few researchers have applied the deep learning algorithms to LfD [10, 26]. Convolutional Neural Networks (CNNs) that are a kind of deep learning models were proposed for image recognition in 1989 [25]. Since CNNs have sparse interactions, parameter sharing, and equivariant representations, they have reduced the model complexity and the computational load. In recent years, CNNs and its different variants applied to a lot of applications in several problem types such as segmentation and prediction outside of classification for both the signal inputs and the image inputs [25]. Recurrent Neural Networks (RNNs) are a kind of artificial neural networks [27]. Since they can perfectly correlate contextual information, they were successfully used in the natural language processing and the modelling and prediction of time series [27]. However, RNNs have the problem of vanishing gradients and exploding gradient obstructing the learning of long data sequences. Long Short Term Memory (LSTM) models were proposed to overcome the problem introducing "gate mechanism" to RNNs to control the flow of *information* [28]. LSTM has a complex structure and high training and testing time [29]. [26] Worked to generate the manipulation trajectories of a 6-axis Lynxmotion AL5D robot with a two-finger gripper by using individually LSTM and CNN in a variational autoencoder structure. In many research areas, the combinations the CNN and LSTM models exhibited showed better performance than CNN and LSTM [30–34]. CNNs extract powerful image features while LSTM models exhibit good results at predicting time series data.

In the current study, the Robotis-Op3 humanoid robot is controlled using a keyboard by a remote human operator and the position information relating to the joints and the images relating to the environment in which the robot looks at are saved [35]. In the proposed LfD system, a new CNN model is firstly introduced as an efficient regressor to imitate the behavior of the human operator. Secondly, the spatial factorization method into asymmetric convolutions in [36] is applied to the proposed CNN model aiming at enhancing the efficiency and effectiveness of the approximation accuracy and reducing the parameter number and obtained new CNN models. Finally, the CNN-LSTM versions of the proposed CNN models are employed to achieve the best estimation accuracy and fast training and testing. Comparative extensive results, the training and validation loss, the parameter number, and the training time and some error measures are illustrated to show the effectiveness of the proposed models.

Table 5 The structure of the Model 5

Type (CNN + LSTM) (3 × 1)(1 × 3)	Output	Filter-Stride	Feature map
Input	360,640	–	3
Conv_1	179,320	5 × 5,2	12
Conv_2	90,159	5 × 5,2	24
MaxPooling_1	45,79	2 × 2,1	24
Conv_3	22,40	5 × 5,2	36
Conv_4	11,19	5 × 5,2	48
MaxPooling_2	5,9	2 × 2,1	48
Dropout	5,9	–	48
TimeDistributed_Flatten_1	5	–	432
LSTM	1	–	36
Dense_1	1	–	100
Dense_2	1	–	50
Dense_3	1	–	30
Dense_4	1	–	1

Table 6 The structure of the Model 6

Type (CNN + LSTM)(5 × 5)(3 × 3)	Output	Filter-Stride	Feature map
Input	360,640	–	3
Conv_1	178,318	5 × 5,2	12
Conv_2	88,158	3 × 3,2	24
MaxPooling_1	44,79	2 × 2,1	24
Conv_3	20,38	5 × 5,2	36
Conv_4	9,18	3 × 3,2	48
MaxPooling_2	4,9	2 × 2,1	48
Dropout	4,9	–	48
TimeDistributed_Flatten_1	4	–	432
LSTM	1	–	36
Dense_1	1	–	100
Dense_2	1	–	50
Dense_3	1	–	30
Dense_4	1	–	1

The rest of this paper is organized as follows: in Sects. 2 and 3 the theoretical background of the CNN and LSTM models are shortly given and introduced proposed CNN and CNN-LSTM models. The comparative experimental results are illustrated to demonstrate the performance of the proposed models in Sect. 4. Section 5 concludes this paper.

2 Theoretical background

This section briefly reviews the fundamental concepts of CNN and LSTM networks.

2.1 Convolutional neural networks

CNNs are a kind of conventional multilayer FNNs [25, 37–39]. CNNs accept the color or gray image data in the form of multidimensional arrays as inputs different from FNNs. Hence, CNNs are mostly used for a lot of applications such as image analysis, video processing, face recognition, object recognition, and natural language processing by using smaller parameter numbers. Thanks to Graphical Processing Units (GPUs), CNNs are applied to big data sets also [25].

CNNs are made of the layers such as the input layer, convolutional layers, nonlinear activation layer, pooling layers, dropout layer, batch normalization layer, one or

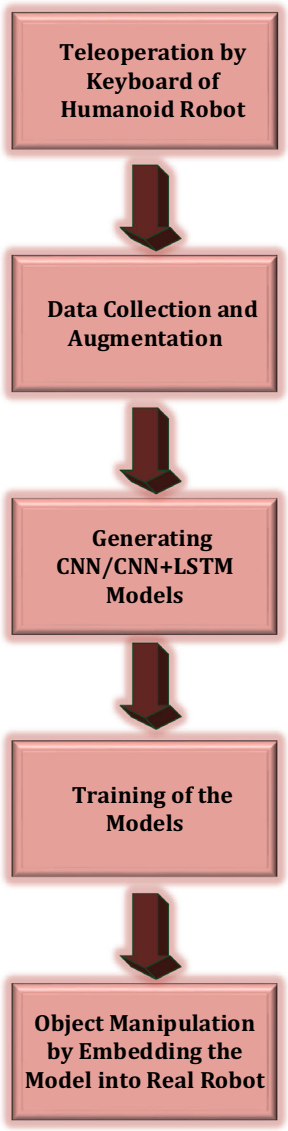


Fig. 4 The framework of learning object manipulation of humanoid robots from the demonstration

multiple fully connected layers, and loss activation layer. A basic structure is depicted in Fig. 1.

CNN performs the operations in the sequential layers defined as below:

- The Input layer: The images consisting of the pixels generate the inputs of CNN.
- Convolution Layer: CNN performs the convolution operation using a set of kernel filters in this layer. The kernel filters extract the features from the given input images by sliding them throughout the images. In the sliding step, the stride parameter is important. Stride shows the number of pixels shifts over the input matrix. When the stride is n constant, the filter is moved n pixels at a time.
- Pooling Layer: CNN performs down-sampling operation along the spatial dimension of the convolution layers in the pooling layer. Thanks to the pooling layer, the feature dimension is reduced without losing any valuable information and so trainable parameter number is reduced. Pooling operation is considered as a solution to the overfitting problem, and vanishing gradient problem. Max pooling operation provides the largest value in the selected region.
- Dropout Layer: CNN performs a regularization operation in the dropout layer. Some nodes of the layer outputs are randomly ignored and temporarily dropped out at the training stage in order to get rid of the overfitting. A flatten layer transforms the features into a vector and it is localized after the dropout layer.
- Batch Normalization Layer: CNN performs the normalization operation of the outputs of the layer outputs. The operation speeds up the training stage and achieves a higher network performance. The output of batch normalization is usually gone through an activation functions such as Rectified Linear Unit (ReLU) defined as max(0, input) to introduce nonlinearity to the data and not to get caught vanishing gradient problem [37].
- Fully Connected Layer: CNN uses the weights to connect every node in the present layer to every node in the next layer similar to FNNs in this layer. The layers perform to the learned features the different activation functions such as softmax and linear with respect to the classification or regression application [25, 37]. The layer is called the dense layer.

2.2 Long short-term memory networks

LSTM network is a different class of deep neural networks including the internal memory loops and the sequences of input data as different from CNNs. Like FNNs, LSTMs are made of an input layer, hidden layer, and output layer [40]. The hidden layer of LSTM consists of a recurrent range of modules including the forget gate, the input gate, and the output gate.

Figure 2 clearly shows a conventional recurrent structure in a recurrent neural network. The demonstration of a LSTM memory cell is shown in Fig. 3. Forget gate, f_t, shaded in blue calculates its output by summing the current input, the outputs of the memory cells at the previous time-step and bias value and then the values importing into σ logistic sigmoid activation function [40]. Hence it determines which information is taken off the cell state, c_t. Input gate, i_t, shaded in yellow, decides which information to be passed through. Output gate, o_t, shaded in green determines which information from the cell state is going to the output. The values $\tilde{c}_t$ in Fig. 2 are the possible nominate values

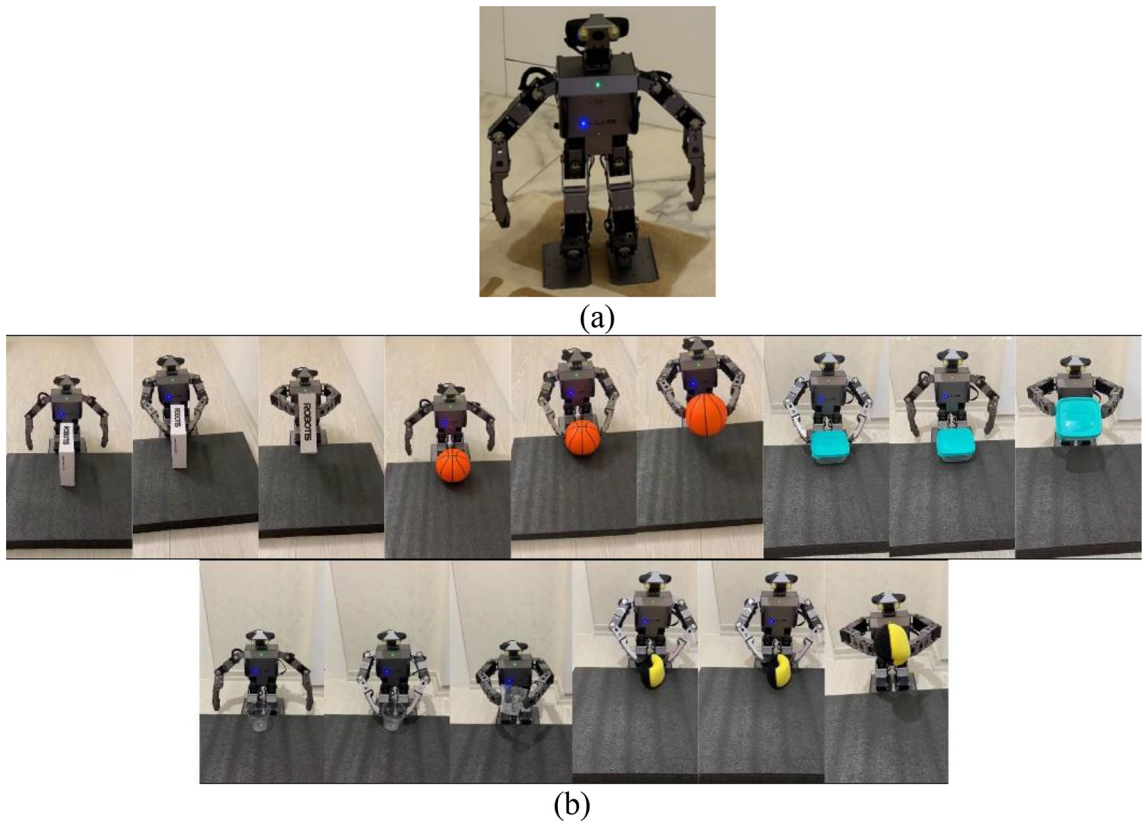


Fig. 5 **a** The Robotis-Op3 humanoid robot and **b** its object manipulation

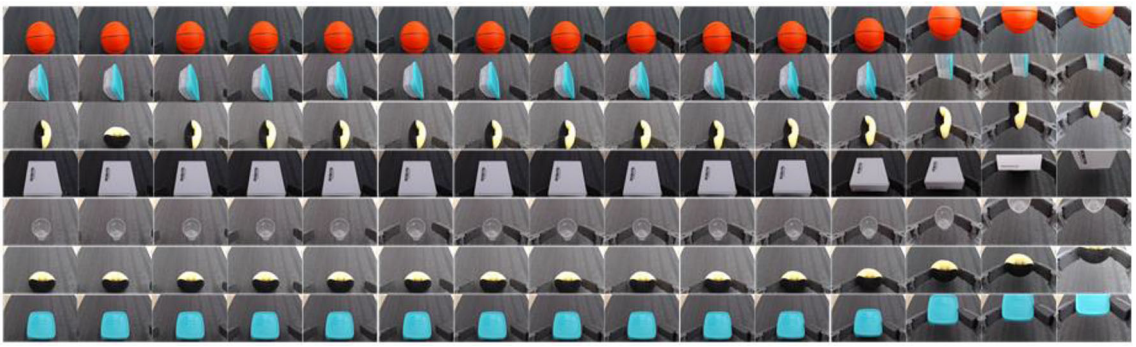


Fig. 6 Some image samples taken from the humanoid robot controlled by keyboard

that are to be added to the cell state. The tanh activation function is used to obtain the values of the nominate cell state $\tilde{c}_t$ in the range of -1 and 1.

Given the sequence of input vector, $x = \{x_1, x_2, \ldots, x_t, \ldots\}$, where $x_t \in \Re^n$ is n-dimensional vector for n variables at time-instance t, the output of LSTM memory cell is calculated as below [41]:

$$i_t = \sigma(w_{ix}x_t + w_{ih}h_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(w_{fx}x_t + w_{hf}h_{t-1} + b_f) \quad (2)$$

$$o_t = \sigma(w_{ox}x_t + w_{oh}h_{t-1} + b_o) \quad (3)$$

$$\tilde{c}_t = \tanh\left(w_{\tilde{c}x}x_t + w_{\tilde{c}h}h_{t-1} + b_{\tilde{c}}\right) \quad (4)$$

$$c_t = f_1 \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (5)$$

$$h_t = o_t \odot \tanh(c_t) \quad (6)$$

where $w_{ix}, w_{ih}, w_{fx}, w_{hf}, w_{ox}, w_{oh}, w_{\tilde{c}x}$, and $w_{\tilde{c}h}$ are weight matrices, b_i, b_f, b_o, and $b_{\tilde{c}}$ are bias vectors. $\odot$ means the Hadamard product of the vectors. h_t is the output vector of the LSTM memory cell in Fig. 3. In LSTM, y_t is the output probability vector of the LSTM as

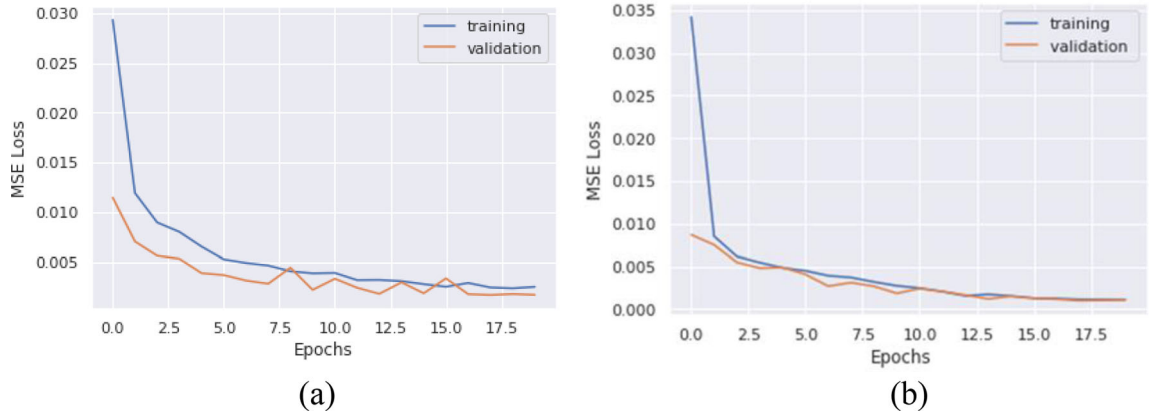


Fig. 7 Training and validation MSE loss values of **a** Model 1 and **b** Model 4 for shoulder roll position of the humanoid robot

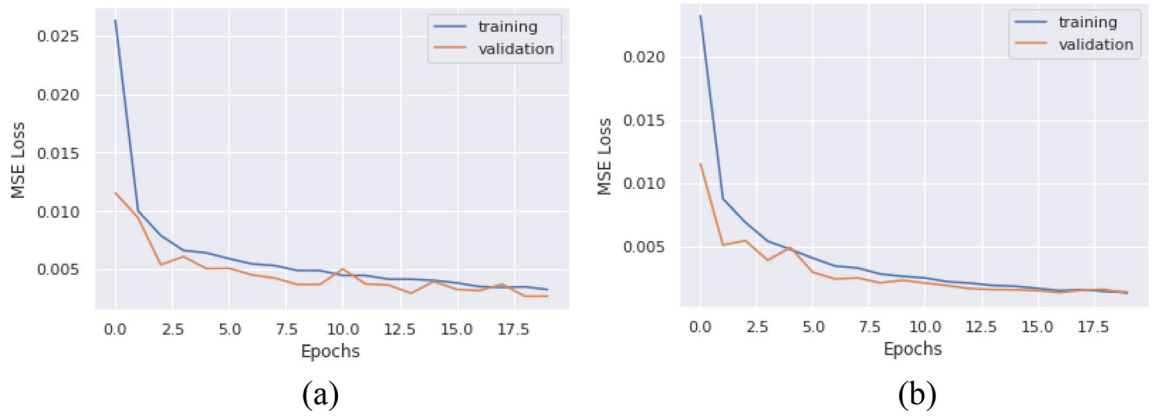


Fig. 8 Training and validation MSE loss values of **a** Model 2 and **b** Model 5 for shoulder roll position of the humanoid robot

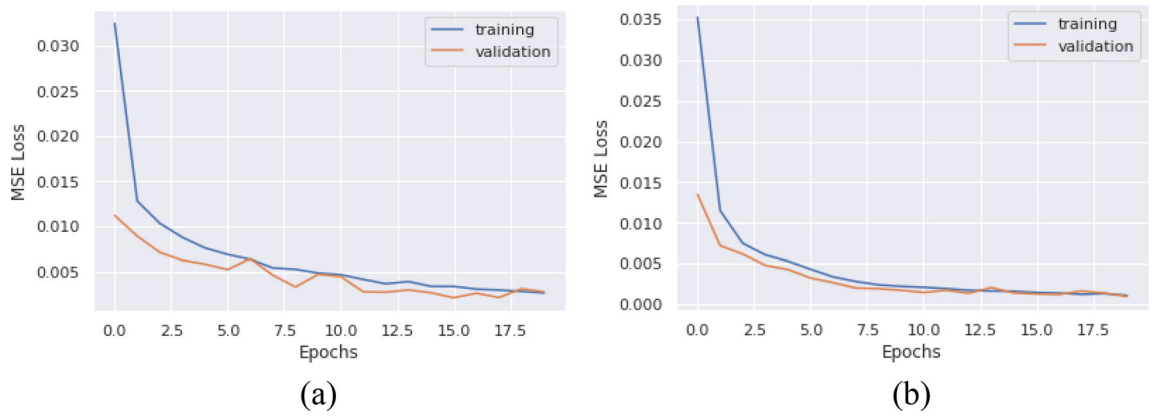


Fig. 9 Training and validation MSE loss values of **a** Model 3 and **b** Model 6 for shoulder roll position of the humanoid robot

$$y_t = w_{yh} h_{t-1} + b_y \tag{7}$$

2.3 Proposed CNN and CNN-LSTM models

The trajectory characteristics of the humanoid robot arm are nonlinear. Hence, the relation between the joint positions and the object to be manipulated should be modelled by a model having a high estimation accuracy. LSTM is suitable to model time-series data but it requires high computational load time and training time. CNN is especially the best at problem types such as object recognition and segmentation that the input is an image. Therefore, this paper proposes a sequential model that

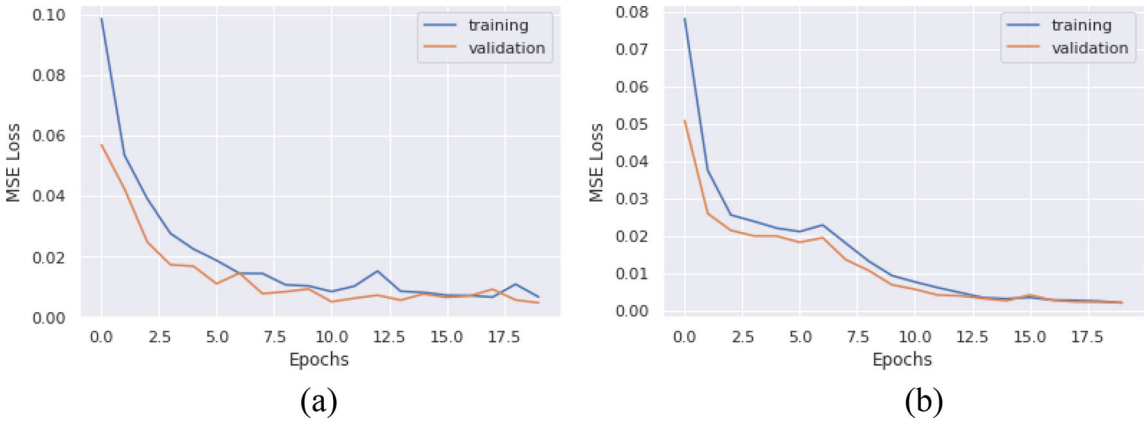


Fig. 10 Training and validation MSE loss values of **a** Model 1 and **b** Model 4 for shoulder pitch position of the humanoid robot

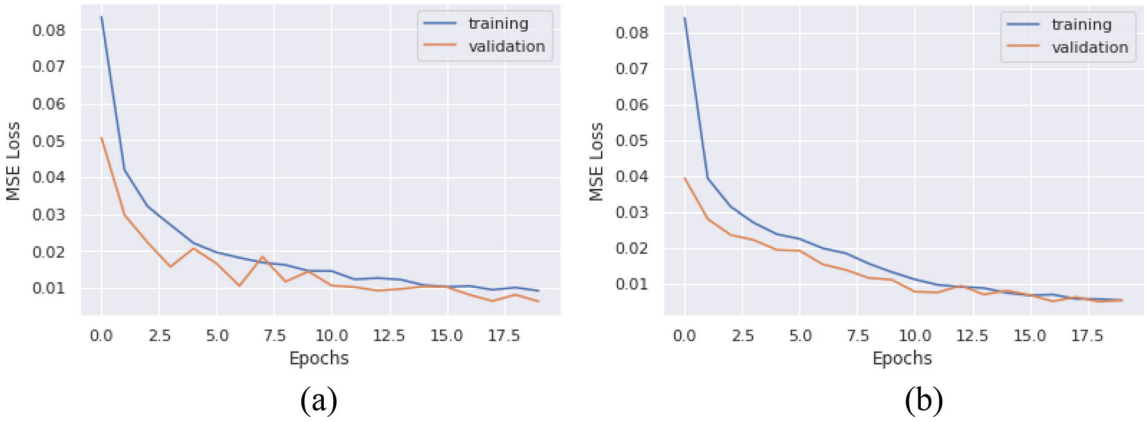


Fig. 11 Training and validation MSE loss values of **a** Model 2 and **b** Model 5 for shoulder pitch position of the humanoid robot

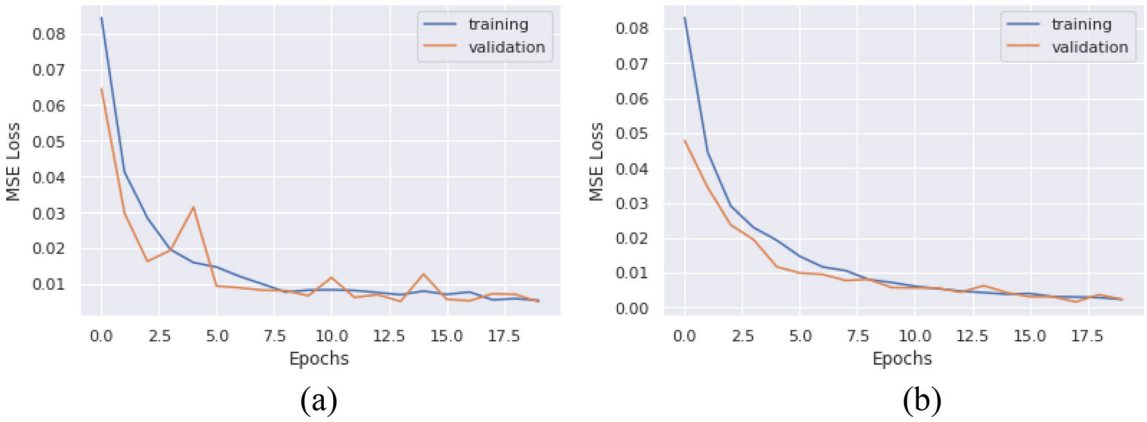


Fig. 12 Training and validation MSE loss values of **a** Model 3 and **b** Model 6 for shoulder pitch position of the humanoid robot

fuses the advantages CNN and LSTM. The sequential model means a structure of ordered and stacked layers. In this structure, CNN layers extract the important features relating to the image and then LSTM and fully connected layers approximate the sequential data.

In this paper, the CNN model in Table 1 is firstly proposed to model the manipulation trajectories inspired by [42]. The table clearly shows the input size, output size, filter size, stride, and the feature map dimension of the model. The proposed model is called as Model 1. Secondly, the scaled versions of Model 1 are proposed to provide less parameter number and better estimation accuracy than Model 1. A scaled procedure is realized as in [36]. Spatial factorization into asymmetric convolutions is applied to

Table 7 The results relating to training/validation loss, parameter number, and training/testing time of the used models for learning the shoulder roll position of the humanoid robot

Models	Training loss	Validation loss	Parameter number	Training time (s)	Testing time (s)
Model 1	0.0025	0.0017	233,331	551.7941	3.3505
Model 2	0.0033	0.0027	231,579	550.3746	3.0464
Model 3	0.0026	0.0028	220,275	549.3001	3.0322
Model 4	**0.0011**	**0.0011**	144,099	552.8487	3.1312
Model 5	0.0014	0.0014	**86,859**	**530.4032**	**2.9666**
Model 6	0.0011	0.0012	118,755	550.7694	3.0154

The bold values show the best values

Table 8 The results relating to training/validation loss, parameter number, and training/testing time of the used models for learning the shoulder pitch position of the humanoid robot

Models	Training loss	Validation loss	Parameter number	Training time (s)	Testing time (s)
Model 1	0.0067	0.0048	233,331	408.0813	3.1689
Model 2	0.0093	0.0064	231,579	407.3395	3.0765
Model 3	0.0054	0.0050	220,275	**404.1772**	3.1035
Model 4	**0.0021**	**0.0021**	144,099	413.1547	3.0936
Model 5	0.0052	0.0052	**86,859**	405.6232	**2.9695**
Model 6	0.0023	0.0024	118,755	408.7509	3.0060

The bold values show the best values

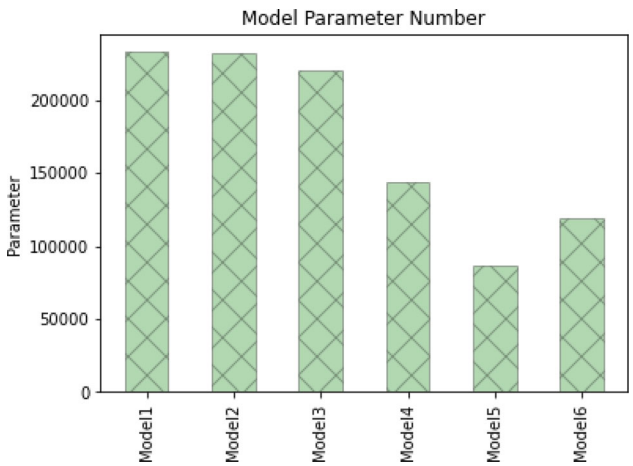


Fig. 13 The parameter numbers of all models

obtain smaller convolutions but high accuracy. In Model 2, (3 × 1) and (1 × 3) scaled convolutions instead of 5 × 5 ones are used. 5 × 5 and 3 × 3 instead of 5 × 5 convolutions are applied in Model 3. The structures of Models 2 and 3 are shown in Tables 2 and 3. Finally, the CNN-LSTM model in Tables 4 and its scaled versions in Tables 5 and 6 are proposed. The models are proposed as Models 5–6, respectively. In the models, the time distributed layers are used to generate a layer for one-time step at a time in LSTM. The CNN-LSTM counterpart of Models 1, 2, 3 in the form of CNN are Models 4 and 6. All models apply the Mean Square Error (MSE) loss function:

$$MSELoss = \frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2 \tag{8}$$

where x is the estimated value, y is the real value, and n is the data number.

3 The proposed algorithm for learning object manipulation of humanoid robots from the demonstration

The framework for learning object manipulation of humanoid robots from demonstration is given in Fig. 4. The steps of the algorithm are as follows:

Step 1: Move the arms of the humanoid robot using a keyboard, which is the teleoperation method, and catch the instances relating to the manipulation environment from the robot camera.

Step 2: Augment the data set to produce the results with high accuracy.

Step 3: Construct the vision-based CNN and CNN-LSTM models to approximate the pitch and roll positions relating to the shoulder of the Robotis-Op3 humanoid robot.

Step 4: Train the models by using the ADAM optimization method [43].

Step 5: Carry out the object manipulation embedding the models into the real robot.

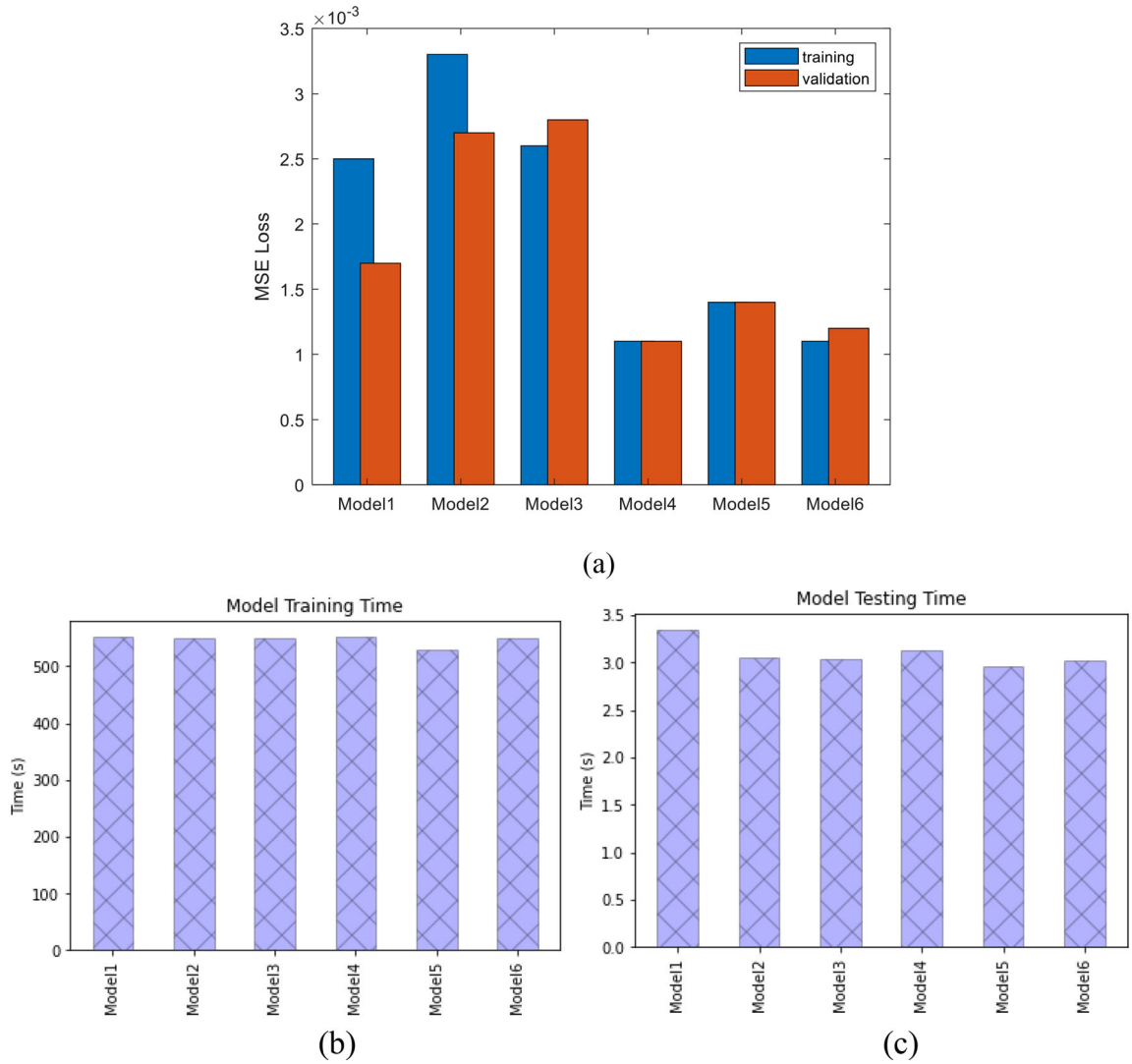


Fig. 14 **a** MSE loss values at the training/validation stage, **b** the training times, and **c** the testing times of the used models for learning the shoulder roll position of the humanoid robot

In the proposed framework, the accuracy of the vision-based LfD system depends on the performance of the CNN/CNN-LSTM models. A large estimation error relating to the model induces larger repeated errors in the object manipulation with the humanoid robots. The reason for why CNN/CNN-LSTM models with different structures are searched in this paper is to catch the manipulation trajectory with high correction in real time in terms of the high approximation performances of CNN and hybrid CNN-LSTM models. The proposed vision method is unsophisticated and easy to apply on the real robots.

4 Experiments

In this paper, six CNN and hybrid CNN-LSTM models are used to learn the object manipulation by applying teleoperation method of LfD on the humanoid robot named as Robotis-Op3 humanoid robot in Fig. 5a [35]. Robotis-Op3 has 20 axes, Intel NUC i3 Dual core 2133 MHz mainboard, 3 axis gyro, 3 axis magnetometer sensors, 3 axis accelerometer, Linux operating system, Logitech C920 HD-Pro camera, C, Robot Operating System (ROS), and Dynamixel SDK. In the experiments, Nvidia Titan XP was used for training the models. The obtained models were embedded into the robot at the ROS environment of the robot and the object manipulation was carried out using Python.

In the experiments, the manipulation of 5 different objects consisting of a storage box, cleaning sponge, glass,

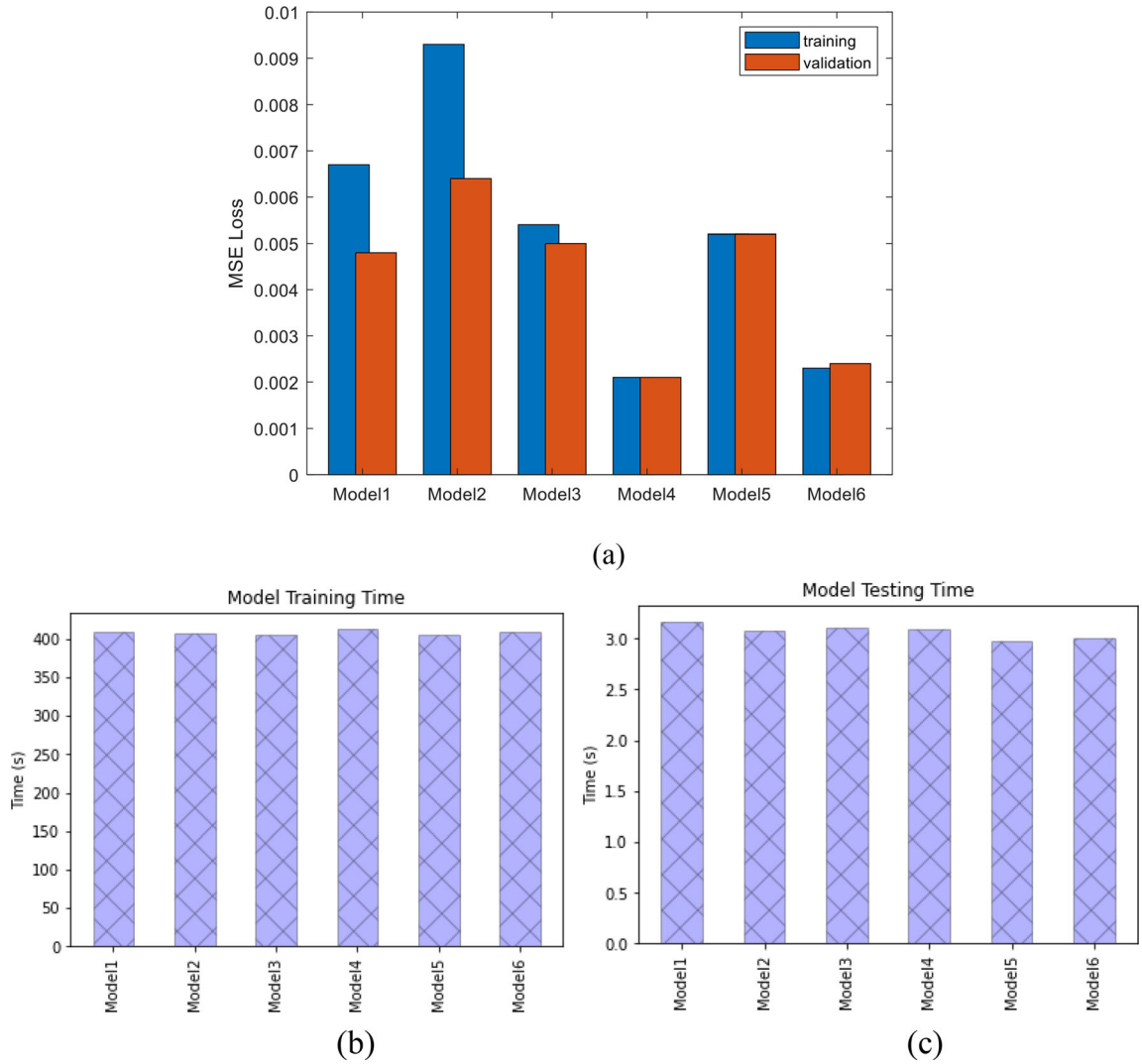


Fig. 15 **a** MSE loss values at the training/validation stage, **b** the training times, and **c** the testing times of the used models for learning the shoulder pitch position of the humanoid robot

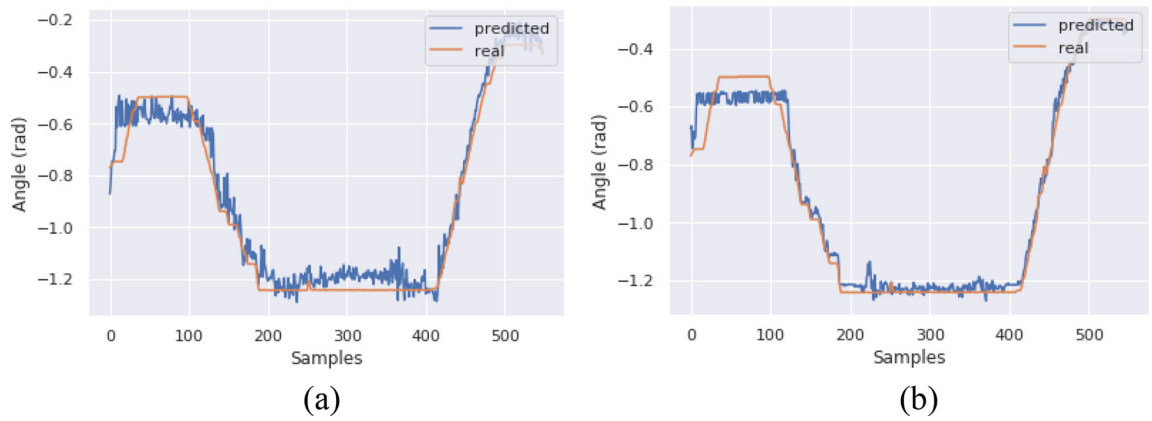


Fig. 16 The predicted and real results of **a** Model 1 and **b** Model 4 for shoulder roll position

ball, and box was carried out as in Fig. 5b. The head and neck angles of the humanoid robot were set to 0, −0.95 rad for the object manipulation scenario in this paper. More complex tasks and scenarios might be approximated

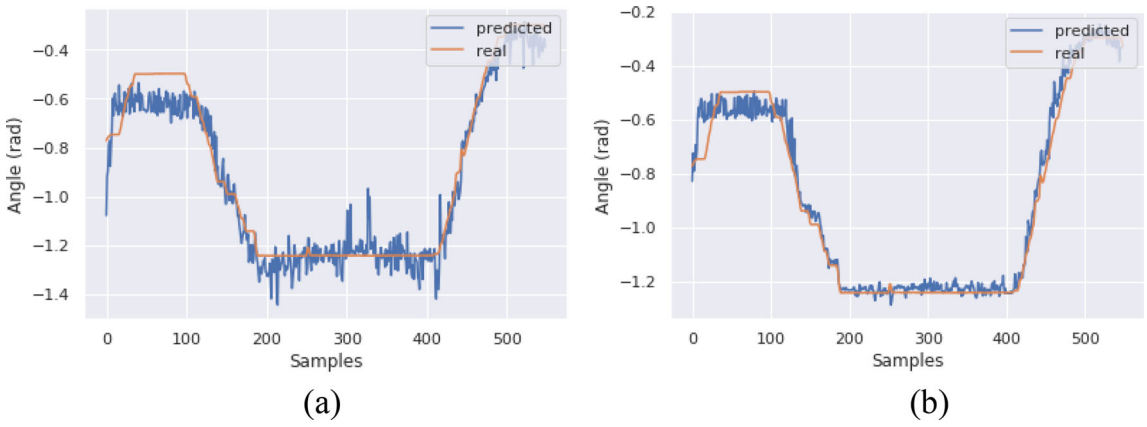


Fig. 17 The predicted and real results of a Model 2 and **b** Model 5 for shoulder roll position

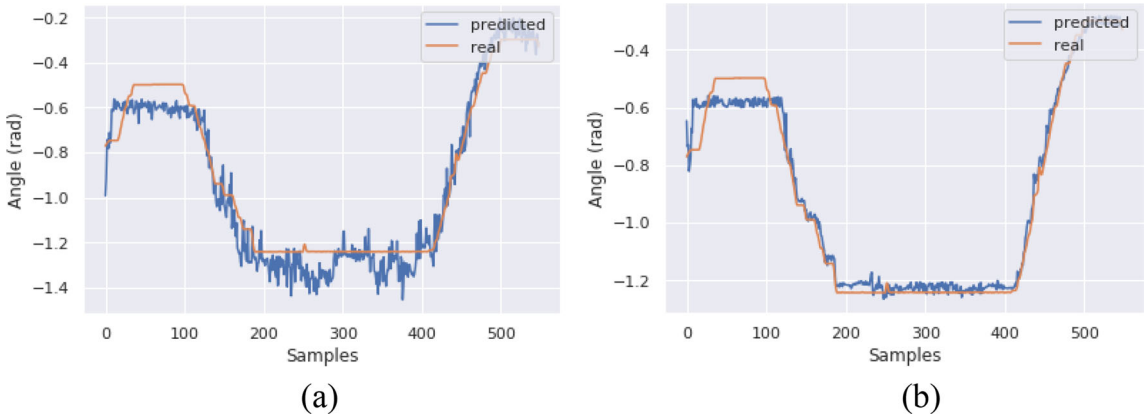


Fig. 18 The predicted and real results of **a** Model 3 and **b** Model 6 for shoulder roll position

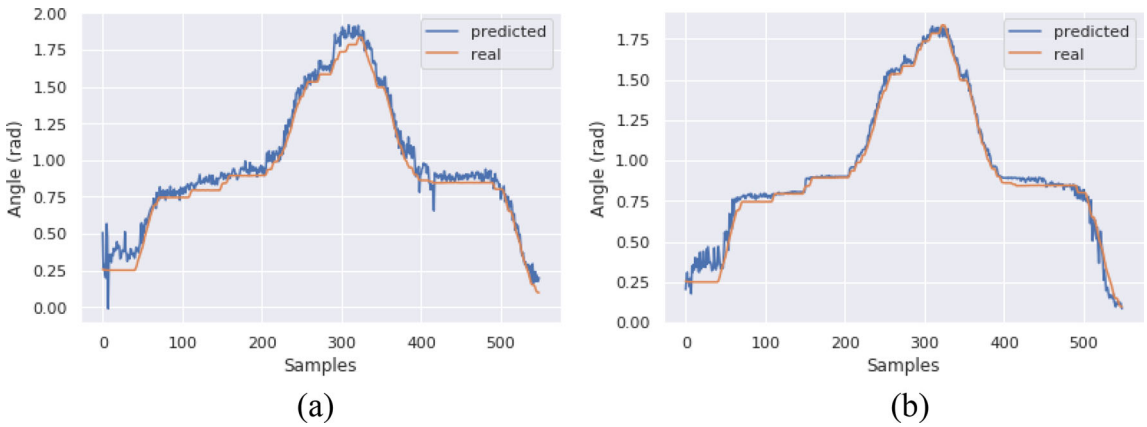


Fig. 19 The predicted and real results of **a** Model 1 and **b** Model 4 for shoulder pitch position

the angles in the future works. The humanoid robot was controlled by a keyboard for perfect manipulation by means of different locations of the objects and multiple demonstrations. While the robot was controlled by the keyboard for object manipulation, the 5102 images with the resolution of 640 × 360 pixels taken from the robot camera and the corresponding joint positions were saved. The first location of the object is on the table. The final location of the object is up and within the hands of the robot. Some sample training images are shown in Fig. 6. Some augmentation methods such as varying the image brightness, rotating the image, and increasing the data having less number with respect to the histogram of joint position distribution were applied to increase the data number

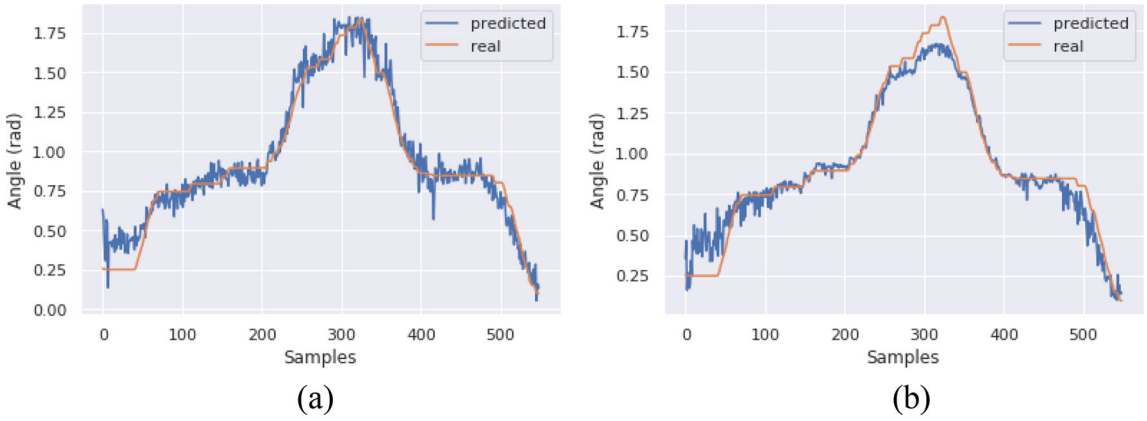


Fig. 20 The predicted and real results of **a** Model 2 and **b** Model 5 for shoulder pitch position

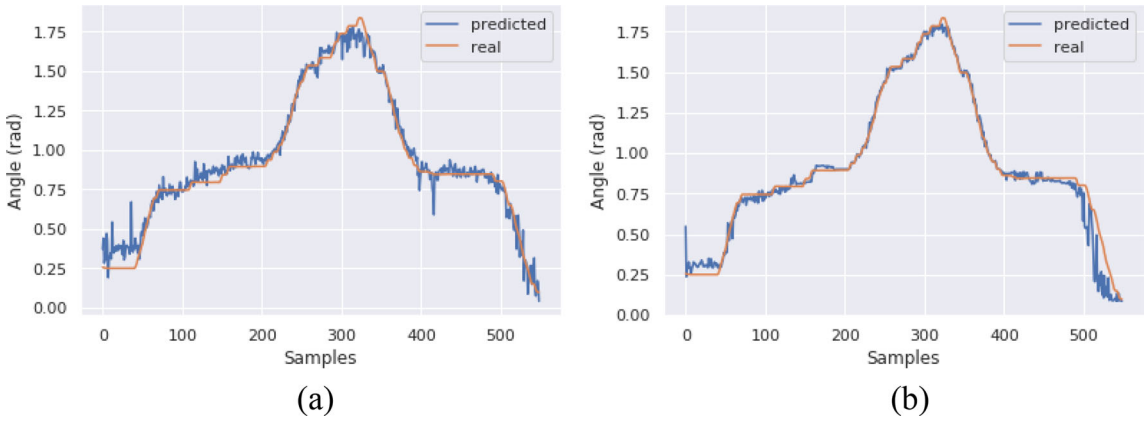


Fig. 21 The predicted and real results of **a** Model 3 and **b** Model 6 for shoulder pitch position

relating to image and position. 80% and %20 of the augmented data set were used for training and validation, respectively. To get rid of overfitting, it was added random distributed noise with approximately 0.001 magnitude to joint positions. All of the models were trained for 20 epochs by using Adam method, MSE loss function, batch size of 8, and steps per epoch of data number/batch size. After training, the models are embedded into the real robot to carry out the object manipulation and the framework is started.

Figures 7, 8, 9, 10, 11 and 12 show the results of the training stage across the epochs for all models. The test stage is realized for the manipulation of an orange ball with the Robotis-Op3 humanoid robot. The training step was worked on multiple times. Tables 7 and 8 show the average results relating to training/validation loss, parameter number, and training/testing time of the used model for learning the shoulder roll and pitch positions of the humanoid robot, respectively. Figure 13 shows the parameter numbers of all models Figs. 14 and 15 show MSE loss values at the training/validation stage, the training times, and the testing times of the used model for learning the shoulder roll and pitch positions of the Robotis-Op3 humanoid robot, respectively.

Tables 7 and 8 list the results relating to training and validation losses, parameter number, and training and testing times of the used models for learning the shoulder roll and pitch positions of the humanoid robot. Figures 7, 8, and 9 show the training and validation MSE loss values of the models with respect to the training epochs for the shoulder roll position. Figures 10, 11, and 12 show training and validation MSE loss values of the models with respect to the training epochs for the shoulder pitch position. Figure 13 shows the parameter numbers of all models. The parameter numbers relating to the Models 1–6 are 233331, 231579, 220275, 144099, 86859, and 118755, respectively. Figures 14 and 15 illustrate MSE loss values and the training and testing times of the models at the training and validation stages for the shoulder roll and pitch positions, respectively. As can be seen from Tables 7 and 8 and Figs. 13, 14, and 15, the proposed Model 1 in the form of CNN results in low training and validation MSE losses. On the other hand, Model 2 and Model 3 that are its factorized variants with $(3 \times 1)(1 \times 3)$ and $(5 \times 5)(3 \times 3)$ depict

the similar results with less parameter number, training time and testing time. In addition, Models 4–6 in the form of CNN-LSTM illustrate lower training and validation losses with smaller parameter number, training time and testing time than their CNN counterparts, Models 1–3, respectively. The best model is Model 4 with the lowest training and validation loss of 0.0011 and 0.0021 for shoulder roll and pitch positions of the humanoid robot, respectively. Considering the smallest parameter number and the smallest testing time, the best one is Model 5 with the testing time with 2.9666 s and the parameter number with 86,859. All results clearly show that the CNN-LSTM models provide more optimum results than the CNN models. Moreover, as can be seen Figs. 7, 8, 9, 10, 11, 12, 14a and 15a, the overfitting risk at the training stage of the CNN-LSTM models are less than the CNN models.

In the testing stage, while the humanoid robot manipulates a ball, the shoulder roll and pitch positions obtained from the CNN models and their counterparts CNN-LSTM models and the real values are comparatively plotted in Figs. 16, 17, 18, 19, 20 and 21. As can be seen from the figures, the CNN-LSTM models achieve the approximation with less noise and deviation. In all models, Model 4 and Model 6 give the best approximation performances. On the other hand, Model 2 provides the worst approximation results including the deviation in higher levels than the others. It is seen that the MSE results in Tables 7 and 8 coincide with the approximation results in Figs. 16, 17, 18, 19, 20 and 21. The low approximation performance is considered as a disadvantage of the models.The performance of our CNN and CNN-LSTM models can be increased by using different optimization methods.

5 Conclusion

In this paper, a new teleoperation framework of LfD was presented using deep learning methods. The new framework successfully transfers the human behavior to the humanoid robot. In the framework, a new CNN model, its scaled versions, and their CNN-LSTM combinations are employed for the object manipulation with humanoid robots.

The main aim of applying CNN and LSTM in the LfD system was to make use of the recognition skill of the CNN models and approximating skill the time series data of the LSTM models and then combine the skills to calculate both fast and correctly the joint positions in the object manipulation with the real humanoid robots.

The LfD system steps consist of the collecting the data relating to joint positions and the images relating to the environment in which the robot saw the moving the arms of humanoid robot by the teleoperation, augmenting the data set to produce the results with high accuracy, constructing the vision-based CNN and CNN-LSTM models to approximate the pitch and roll positions relating to the shoulder of the Robotis-Op3 humanoid robot and finally realizing the object manipulation embedding the trained model into the real robot.

A comparative study was built for evaluating the proposed model performances. The experimental results showed that the proposed CNN and CNN-LSTM models can approximate the nonlinear characteristics of object manipulation. On the other hand, the scaled variants provided similar results to their counterparts with small parameters. Moreover, compared with CNN models, the proposed CNN-LSTM models presented smoother estimation results and better tracking accuracy in the test stage. Besides, the proposed CNN-LSTM models well learned the joint positions with 0.001 and 0.002 MSE losses even under random distributed noise in a faster way than the others.

In addition, it is notable to state that the proposed models can also be applied to other regression problems in real life applications. Next studies will be designed based on the deep reinforcement learning by using the models.

Acknowledgements This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) grant numbers 117E589. In addition, GTX Titan X Pascal GPU in this research was donated by the NVIDIA Corporation.

Author contributions SNA: Software, Methodology, Conceptualization, Investigation, Writing. RÖ: Software, Methodology, Validation, Formal analysis. AU: Conceptualization, Software, Writing—Review & Editing, Supervision, Writing—Original draft preparation, Project administration. CG: Conceptualization, Supervision.

Funding This work was funded by the Scientific and Technological Research Council of Turkey (TUBITAK) grant numbers 117E589. In addition, GTX Titan X Pascal GPU in this research was donated by the NVIDIA Corporation.

Data Availability The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

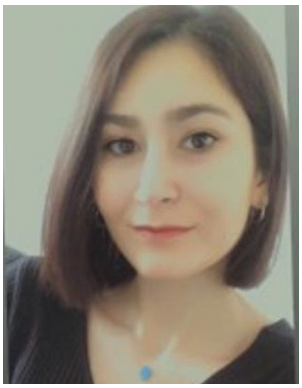
Ethical Approval For this type of study formal consent is not required.

Informed consent Additional informed consent was obtained from all individual participants for whom identifying information is included in this article.

References

1. Schaal, S.: Learning from demonstration. In: Advances in Neural Information Processing Systems, vol. 9. MIT Press, pp. 1040–1046 (1997)
2. Argall, B.D., Chernova, S., Veloso, M., Browning, B.: A survey of robot learning from demonstration. Robot. Auton. Syst. **57**, 469–483 (2009)
3. Billard, A.G., Calinon, S., Dillmann, R.: Learning from humans. In: Springer Handbook of Robotics, pp. 1995–2014. Springer, Berlin (2016)
4. Akgun, B., Cakmak, M., Jiang, K., Thomaz, A.L.: Keyframe-based learning from demonstration. Int. J. Soc. Robot. **4**, 343–355 (2012)
5. Fischer, K., Kirstein, F., Jensen, L.C., Krüger, N., Kukliński, K., aus der Wieschen, M.V., Savarimuthu, T.R.: A comparison of types of robot control for programming by demonstration. In: 2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI). pp. 213–220. IEEE (2016)
6. Praveena, P., Subramani, G., Mutlu, B., Gleicher, M.: Characterizing input methods for human-to-robot demonstrations. In: 2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI). pp. 344–353. IEEE (2019)
7. Akguen, B., Subramanian, K., Thomaz, A.L.: Novel interaction strategies for learning from teleoperation. In: AAAI Fall Symposium: Robots Learning Interactively from Human Teachers. p. 07 (2012)
8. Lee, D., Ott, C.: Incremental motion primitive learning by physical coaching using impedance control. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 4133–4140. IEEE (2010)
9. Saveriano, M., An, S., Lee, D.: Incremental kinesthetic teaching of end-effector and null-space motion primitives. In: 2015 IEEE International Conference on Robotics and Automation (ICRA). pp. 3570–3575. IEEE (2015)
10. Ravichandar, H., Polydoros, A.S., Chernova, S., Billard, A.: Recent advances in robot learning from demonstration. Annu. Rev. Control Robot. Auton. Syst. **3**, 297–330 (2020)
11. Hersch, M., Guenter, F., Calinon, S., Billard, A.G.: Learning dynamical system modulation for constrained reaching tasks. In: 2006 6th IEEE-RAS International Conference on Humanoid Robots. pp. 444–449. IEEE (2006)
12. Abu-Dakka, F.J., Huang, Y., Silvério, J., Kyrki, V.: A probabilistic framework for learning geometry-based robot manipulation skills. Robot. Auton. Syst. **141**, 103761 (2021)
13. Al-Yacoub, A., Zhao, Y.C., Eaton, W., Goh, Y.M., Lohse, N.: Improving human robot collaboration through Force/Torque based learning for object manipulation. Robot. Comput. Integr. Manuf. **69**, 102111 (2021)
14. Angelov, D., Hristov, Y., Ramamoorthy, S.: Using causal analysis to learn specifications from task demonstrations. arXiv preprint https://arxiv.org/abs/1903.01267 (2019)
15. Yavşan, E., Uçar, A.: Gesture imitation and recognition using Kinect sensor and extreme learning machines. Measurement **94**, 852–861 (2016)
16. Meccanici, F.: Teleoperated online learning from demonstration in a partly unknown environment: using a semiautonomous care robot. MS Thesis. Delft University of Technology (2021)
17. Lee, D., Choi, H., Chung, W.K., Kim, K.: Arc-length based two-step robot motion teaching method for dynamic tasks. In: 2020 17th International Conference on Ubiquitous Robots (UR). pp. 17–22. IEEE (2020)
18. Tripathi, U., Saran, R., Chamola, V., Jolfaei, A., Chintanpalli, A.: Advancing remote healthcare using humanoid and affective systems. IEEE Sens. J. (2021). https://doi.org/10.1109/JSEN.2021.3049247.
19. Mueller, C., Venicx, J., Hayes, B.: Robust robot learning from demonstration and skill repair using conceptual constraints. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 6029–6036. IEEE (2018)
20. Si, W., Wang, N., Yang, C.: A review on manipulation skill acquisition through teleoperation-based learning from demonstration. Cogn. Comput. Syst. **3**, 1–16 (2021)
21. Xie, Z., Zhang, Q., Jiang, Z., Liu, H.: Robot learning from demonstration for path planning: a review. Sci. China Technol. Sci. (2020). https://doi.org/10.1007/s11431-020-1648-4
22. Jain, A., Wojcik, B., Joachims, T., Saxena, A.: Learning trajectory preferences for manipulators via iterative improvement. arXiv preprint https://arxiv.org/abs/1306.6294 (2013)
23. Basu, C., Singhal, M., Dragan, A.D.: Learning from richer human guidance: augmenting comparison-based learning with feature queries. In: Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction. pp. 132–140 (2018)
24. Chao, C., Cakmak, M., Thomaz, A.L.: Towards grounding concepts for transfer in goal learning from demonstration. In: 2011 IEEE International Conference on Development and Learning (ICDL). pp. 1–6. IEEE (2011)
25. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436 (2015). https://doi.org/10.1038/nature14539
26. Rahmatizadeh, R., Abolghasemi, P., Bölöni, L., Levine, S.: Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration. In: 2018 IEEE international conference on robotics and automation (ICRA). pp. 3758–3765. IEEE (2018)
27. Connor, J.T., Martin, R.D., Atlas, L.E.: Recurrent neural networks and robust time series prediction. IEEE Trans. Neural Netw. **5**, 240–254 (1994)
28. Hochreiter, S., Schmidhuber, J.: LSTM can solve hard long time lag problems. In: Advances in Neural Information Processing Systems, vol. 9. MIT Press, pp. 473–479 (1997)
29. Shewalkar, A.N.: Comparison of RNN, LSTM and GRU on speech recognition data. MS Thesis, North Dakota State. University (2018)
30. Li, T., Hua, M., Wu, X.: A hybrid CNN-LSTM model for forecasting particulate matter (PM2. 5). IEEE Access **8**, 26933–26940 (2020)
31. Yan, R., Liao, J., Yang, J., Sun, W., Nong, M., Li, F.: Multi-hour and multi-site air quality index forecasting in Beijing using CNN, LSTM, CNN-LSTM, and spatiotemporal clustering. Expert Syst. Appl. **169**, 114513 (2021)
32. Livieris, I.E., Kiriakidou, N., Stavroyiannis, S., Pintelas, P.: An advanced CNN-LSTM model for cryptocurrency forecasting. Electronics **10**, 287 (2021)
33. Liu, Z., Zhang, D., Luo, G., Lian, M., Liu, B.: A new method of emotional analysis based on CNN–BiLSTM hybrid neural network. Clust. Comput. **23**, 2901–2913 (2020)
34. Aslan, S.N., Ozalp, R., Uçar, A., Güzeliş, C.: End-to-end learning from demonstation for object manipulation of robotis-Op3 humanoid robot. In: 2020 International Conference on INnovations in Intelligent SysTems and Applications (INISTA). pp. 1–6. IEEE (2020)
35. Robotis-Op3.: http://emanual.robotis.com/docs/en/platform/op3/introduction/ (2020). Accessed 5 May 2020
36. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826 (2016)
37. Srinivas, S., Sarvadevabhatla, R.K., Mopuri, K.R., Prabhu, N., Kruthiventi, S.S., Babu, R.V.: A taxonomy of deep convolutional neural nets for computer vision. Front. Robot. AI **2**, 36 (2016)

38. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Adv. Neural. Inf. Process. Syst. **25**, 1097–1105 (2012)
39. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015)
40. Olah, C.: Understanding lstm networks (2015). https://colah.github.io/posts/2015-08-Understanding-LSTMs/
41. Graves, A., Jaitly, N., Mohamed, A.: Hybrid speech recognition with deep bidirectional LSTM. In: 2013 IEEE workshop on automatic speech recognition and understanding. pp. 273–278. IEEE (2013)
42. Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J.: End to end learning for self-driving cars. arXiv preprint https://arxiv.org/abs/1604.07316 (2016)
43. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint https://arxiv.org/abs/1412.6980 (2014)



Simge Nur Aslan received a BS degree from the Mechatronics Engineering Department at the University of Firat of Turkey in 2019, respectively. He is currently a master student in the same department. His research interests include humanoid robots and deep learning.

Recep Özalp received a BS degree and MS degree from the Mechatronics Engineering Department at the University of Firat of Turkey in 2018 and 2020, respectively. He is currently a Ph.D student in the same department. His research interests include humanoid robots, FPGA, and deep learning.

Ayşegül Uçar received a BS degree, MS degree, and PhD degree from the Electrical and Electronics Engineering Department at the University of Firat of Turkey in 1998, 2000, and 2006, respectively. In 2013, she was a visiting professor at Louisiana State University in the USA. She has been a professor in the Department of Mechatronics Engineering since 2020. She has more than 21 years background in autonomous technologies and artificial intelligence, its engineering applications, robotics vision, teaching and research. Ucar is active in several professional bodies, in particular, she is an associate editor of IEEE Access and Turkish Journal Electrical Engineering and Computer Sciences and a member of European Artificial Intelligence Alliance Committee.

Cüneyt Güzeliş received the B.Sc., M.Sc., and Ph.D. degrees in electrical engineering from Istanbul Technical University, Istanbul, Turkey, in 1981, 1984, and 1988, respectively. He was with Istanbul Technical University from 1982 to 2000 where he became a full professor. He worked between 1989 and 1991 in the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, California, as a visiting researcher and lecturer. He was with the Department of Electrical and Electronics Engineering from 2000 to 2011 at Dokuz Eylül University, Izmir, Turkey. He was with Izmir University of Economics, Faculty of Engineering and Computer Sciences, Department of Electrical and Electronics Engineering from 2011 to 2015. He is currently working in Yaşar University, Faculty of Engineering, Department of Electrical and Electronics Engineering. His research interests include artificial neural networks, biomedical signal and image processing, nonlinear circuits-systems, and control, and educational systems.