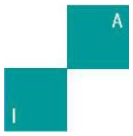
Inteligencia Artificial 22(63), 121-134
doi: 10.4114/intartif.vol22iss63pp121-134



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Users Activity Gesture Recognition on Kinect Sensor Using Convolutional Neural Networks and FastDTW for Controlling Movements of a Mobile Robot

Miguel Pfitscher[1], Daniel Welfer[2], Evaristo José do Nascimento[3], Marco Antonio de Souza Leite Cuadros[4] and Daniel Fernando Tello Gamarra[5]
[1,2,3,5] Universidade Federal de Santa Maria (UFSM)
Santa Maria RS 97105-900, Brazil
miguel.pfitscher@gmail.com, welfer@gmail.com, evaristo.nascimento@ecomp.ufsm.br, daniel.gamarra@ufsm.br
[4] Instituto Federal do Espirito Santo
Serra, ES 29173-087, Brazil
marcoantonio@ifes.edu.br

Abstract In this paper, we use data from the Microsoft Kinect sensor that processes the captured image of a person using and extracting the joints information on every frame. Then, we propose the creation of an image derived from all the sequential frames of a gesture the movement, which facilitates training in a convolutional neural network. We trained a CNN using two strategies: combined training and individual training. The strategies were experimented in the convolutional neural network (CNN) using the MSRC-12 dataset, obtaining an accuracy rate of 86.67% in combined training and 90.78% of accuracy rate in the individual training. Then, the trained neural network was used to classify data obtained from Kinect with a person, obtaining an accuracy rate of 72.08% in combined training and 81.25% in individualized training. Finally, we use the system to send commands to a mobile robot in order to control it.

Keywords: Human gestures recognition, convolutional neural networks, Microsoft Kinect, MSRC-12 dataset, Mobile robot.

1 Introduction

Human physical activity recognition from skeleton data has attracted increasing attention in signal and image processing due to the variety of applications in which it could be used. The process of recognizing human actions implies processing a sequence of frames, looking one frame at a time. In order to help to solve this problem the emergence of a new generation of optimized frameworks has made Convolutional Neural Networks (CNN) popular and efficient for solving many problems in image recognition. In this paper, we present a way to use CNN to recognize gestures and human actions more efficiently in order to use them in real time. We also present a comparison of two forms of training for the CNN structure.

The Kinect sensor from Microsoft processes, collects and recognizes human joints. This processing facilitates the recognition of actions, the collected joints information is processed to facilitate gesture recognition. In the other hand, different databases have been created using the Kinect sensor in order to test different machine learning algorithms to recognize human gestures or activities for different purposes. Mo et al. [1] uses the dataset CAD-60, otherwise, in this article, we used the MSRC-12 dataset [2] for training and evaluation, the referred dataset has 6244 gesture instances of 12 actions. In order to make that all the gestures of the dataset could have fixed size of frames, a Fast-dynamic time warping (FastDTW) algorithm was used [3]. After training the neural

ISSN: 1137-3601 (print), 1988-3064 (on-line)

network, we used a Kinect sensor to collect data and, thus, it is possible to make that the employed network could recognize the gestures of a person and control a mobile robot.

The article is divided in seven sections, after a brief introduction, the second section explores some related works, the third section is a short summary about convolutional neural networks, the fourth section explains the Fast Dynamic time warping algorithm, the fifth section describes the experimental setup, and the sixth section depicts the method proposed in this work, the seventh section shows the obtained results, and finally, conclusions are summarized in the last section.

2 Related Work

In this section, we review some related works for skeleton-based action detection, and also papers that address the robot control problem using the Kinect sensor.

2.1 Deep Learning for Gesture Recognition

Mo et al. [1] used a computer vision model based on the deep learning algorithm to recognize human physical activity from the Microsoft Kinect. The skeleton data from the CAD-60 dataset was used for training and testing. First, the data is processed and prepared to be use in the deep learning algorithm. The CAD data set was labelled and grouped in small sets of 48 frames. Thus, it was possible to get approximately 3500 samples for training and evaluation. The model uses a convolutional neural network and a multilayer perceptron to classify twelve human activities. Moreover, the model structure has an input data size of 144x48 and the architecture of the network alternates three convolutional layers with three pooling layers. Finally, a multi-layer perceptron was used to generate the output. As a result, an accuracy of 81.8% is obtained on the validation set.

Hou et al. [4] proposed a structure using convolutional neural networks to recognize human actions, where three datasets were used for training and evaluation, namely: the MSRC-12 Kinect Gesture, the G3D and the UTD-MHAD datasets. Each action, in this skeleton dataset, was divided in three scatter plots, which creates spectral distributions of the joints. Each of the three spectral distributions creates an image which is used to train the neural network. Thus, they have three outputs scores that will be fused to get a score. So, the process keeps the basic spatial information, but the temporal information is lost [4]. As seen before, MSRC-12 Kinect Gesture dataset is a relatively large data set for action recognition. It contains 594 sequences with 12 gestures, 6244 gestures instances in total.

Jiang et al. [5] show a new approach for human action recognition. The main difference between this approach and a traditional method is that Jiang divides each action into some human segments as pre-processing and then use a k-nearest neighbours (KNN) classifier to classify each group. This approach can be better for handling complex motion variations. After segmentation, the data is classified in motion vectors and for online recognition the approach is similar with the addition of spatial temporal features. As a result, this model achieved at least 90% for every gesture using the MSRC-12 dataset.

Ke et al. [6], considered a CNN model for 3-D action recognition. The joints information is transformed into images, and feed to the deep learning network. Instead of treating the features of all frames as a time series, it is generated a discriminative and compact representation for action recognition to learn robust temporal information. Using NTU RGB+D Dataset, they got an accuracy of 75.9%.

Sharaf et al. [7] proposed the use of a support vector machine (SVM) to recognize human activities in real-time from 3D skeleton data. They used multivariate statistical methods for encoding the relationship between the extracted features. Besides, it was proposed a multi-scale action detector to process a sequence of frames at different scales.

Nguyen and Le [8], demonstrate that relevance vector machines (RVMs) can also achieve a good performance. They use the MSRC-12 data set, obtaining an accuracy of 93.6%. Pan and Li [9], adapt the dynamic time warping (DTW) algorithm to reduce the pathological alignment in resource extraction and model generation, caused by the traditional DTW. This algorithm was used for gesture recognition in the MSRC-12 data set. They get a hit rate of 75.1%.

Other approaches that have also been explored for the problem of gesture recognition are the Hidden Mark Models as in the work of Xia et al. [10] and Piyathilaka et al. [11]; multi kernel as in the work of Althloothi et al. [12]; or the use of hierarchical Recurrent Neural Networks (RNN) as in the work of Du [13]; Oyedotun et al. [14] used Convolutional Neural Networks to recognize static hand gesture.

2.2 Gesture Recognition with Kinect for the Control of Mobile Robots

We also review papers that address the robot control problem using the Kinect sensor. Among these works, we could mention the work developed by Borja et al. [15] that presented an algorithm which makes tracking of the hand using just images of depth captured with a Kinect sensor, it makes them invariant to light and skin colour conditions. To obtain the Kinect images, the OpenNi library was used. They use this tracking to control the speed and angular position of a mobile robot. The algorithm uses the previous frame for the segmentation of the current one, thus, a user must extend the hand in front of the camera and leave it for a while until the program recognizes the hand. Also, Borja, design a PID control for controlling the wheel speed of the robot.

Wang et al. [16] propose a simple method to control the movement of a robot using Kinect skeleton data. They used the coordinate of joints, which were obtained by Kinect SDK to make a gesture recognition of eleven simple gestures, and, then, control the movements of a Khepera III robot. For gesture recognition, they used a method based on angles of each gesture.

Zhao et al. [17] show a calling gesture recognition for taking order service of an elderly care robot. It was designed mainly for helping non-expert users like elderly to call a service robot. They used a skeleton based gesture recognition and, also, an Octree based gesture recognition. This method was implemented on a service robot developed for elderly care.

Limin et al. [18] used a Kinect camera to track the human skeleton points and capture human actions in real time. They used this information to send commands to the robot through the Bluetooth communication and make some movements as turning and forward.

Fadli et al. [19] proposed a system which allows us to instruct a robot to imitate what we are doing. They used a Kinect for capture information about its skeleton model and, then, a humanoid robot gets commands to move based on obtained angle data and imitate a body posture.

Other works applying the gesture recognition with the Kinect sensor with a mobile robot or manipulator robot have been use, such in the work of Kundu et al. [20], that propose an omnidirectional drive system to detect and pass a port; The work of AbdelKrim et al. [21], which show a reactive navigation system for internal environment using the Kinect sensor; Kameyama and Hidaka [22] which demonstrate an autonomous exploration algorithm of unknown environment for mapping; and Bellarbi et al. [23], that show a navigation method on a mobile robot and a human-robot interaction.

3 Theoretical Background

3.1 Convolutional Neural Networks (CNNs)

Deep learning is a subfield of machine learning, which uses hierarchical architectures to attempt to learn high levels of data abstraction without the need to detail how the algorithm will work or explicitly describe the characteristics of the data or the solution required. The deep learning technique is defined by a neural network architecture composed of several layers, and every layer has neurons (i.e. processing units) in their structure. Convolutional Neural Networks is one of the most popular categories of deep learning neural networks for being efficient in image recognition. As referred in Guo et al [24] It has demonstrated a high accuracy in diverse computer vision applications. This neural network mainly uses three types of layers, a convolutional layer, a pooling layer and a fully connected layer. Generally, the convolutional layers and the pooling layers are interleaved and the use of the full connected layers is limited to the end of the neural network. Figure 1 shows an example of the architecture of a deep convolutional network where we can see the input, pooling or convolutional layers, the fully connected layers and the output layer.

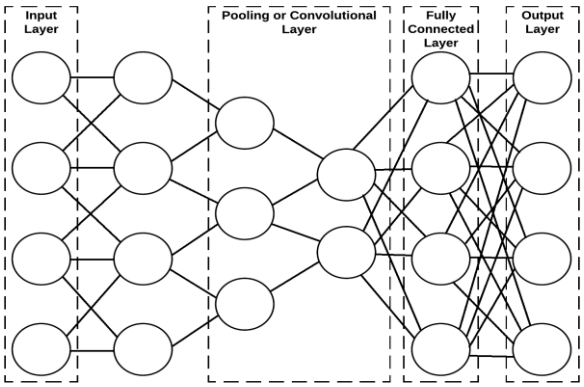


Figure 1: CNN architecture.

3.1.1 Convolutional Layer

This layer executes the operation of convolution of the feature maps of the input layer or those of previous layers using filters; each filter is used to detect a type of feature. This layer has a kernel with two dimensions that corresponds to the size of the convolution window, that is, we can size the kernel according to the size of the feature that we want to map. There are three main advantages of the convolution operation: 1) the weight sharing mechanism in the same feature map reduces the number of parameters 2) The local connectivity learns correlations among neighbouring pixels 3) The invariance to the location of the object as referred in [24].

3.1.2 Pooling Layer

The pooling layers are simple, but really useful to reduce the size of the feature maps, for this, it usually follows a convolutional layer. A pooling layer with the kernel size of 2×2, for example, reduces a path with 2×2 pixels to one pixel, choosing the maximum pixel or the average pixel for max-pooling or average-pooling respectively. it is easy to infer that the convergence of max-pooling is faster than that of average pooling which makes max-pooling to be used in many applications.

3.1.3 Fully Connected Layer

Fully-connected layers are widely used at the end of CNNs to reshape the previous layer (usually 2D) to a single dimension. The fully-connected layers contain about 90% of the total parameters in a CNN and are responsible for most of the training computational cost [25].

3.2 The Fast Dynamic Time Warping Algorithm (FastDTW)

Dynamic time warping (i.e. DTW) is a technique that finds the optimal alignment between two time series, if one time series may be “warped” non-linearly by stretching or shrinking it along its time axis as referred in [3]. Chu et al. [26] show that this similarity measurement can be computationally expensive. A traditional DTW has a time and spatial complexity of O (N^2) which make it considerably time consuming to execute for each gesture of a large dataset. Thus, we use an algorithm that is an approximation of DTW, the algorithm used was the FastDTW [3], an algorithm that performs well with a O (N) time and memory complexity. the algorithm uses three main treatments: Coarsening, which is responsible to decrease a time series into a smaller one that represents the same curve with fewer data points; Projection is used to find a minimum distance warp path at a lower resolution, and use it as an initial guess for a higher resolution warp path; Refinement, by locally adjusting the warp path, it will refine the warp path projected from a lower resolution.

The FastDTW algorithm uses a multilevel graph bisection algorithm, which will split a graph and get smaller graphs as possible. This multilevel approach is used to find an optimal solution for each small graph and makes

the algorithm linear in time and space, as shown by Stan [3]. Defining two time series as X and Y, equation (1) and (2) respectively, where x_k and y_k are each time series element:

$$X = x_1, x_2, \ldots, x_k \quad (1)$$

$$Y = y_1, y_2, \ldots, y_k \quad (2)$$

and constructing a wrap path as W given by equation (3):

$$W = w_1, w_2, \ldots, w_k \; max\,(|X|, |Y|) \leq K < |X| + |Y| \quad (3)$$

An optimal warp path is the minimum distance warp path, where the distance of a warp path W is given in equation (4), where Dist(w_{ki}, w_{kj}) is the distance between the two data point indexes (i.e. one from X and the other one from Y).

$$\text{Dist (W)} = \sum_{k=1}^{k=K} \text{Dist}(w_{ki}, w_{kj}) \quad (4)$$

4 Experimental Setup

4.1 The MSRC-12 Dataset

In this article, we used the MSRC-12 dataset [2] which has 6244 gesture instances of 12 actions. These actions can have a variable number of frames and that could make it hard to train in a fixed neural network with a fixed-size input. The dataset contains 594 sequences in 719359 frames that were collected from 30 different people performing 12 actions, providing in total 6244 gestures. We can see all the gestures obtained from this dataset in Figure 2.

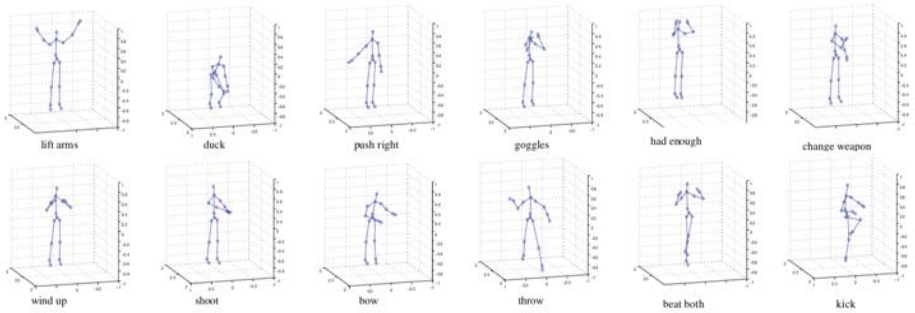


Figure 2: Gestures in MSRC-12 dataset.

4.2 Sensor Kinect

The Kinect sensor is a device manufactured by Microsoft with the intention of collecting data in RGBD for the purpose of using them in the recognition of the most diverse human movements. It was developed specifically for the Xbox videogame which brought greater interactivity and immersion into your games. Kinect consists of some sensors that make it possible to capture data from people, or objects mapping the environment in three dimensions. This mapping is due to the set of sensors: a camera, where the image data is obtained; An infrared (IR) projector, which works to obtain depth data from the site; An IR camera, which, along with the infrared projector, get the depth data being read by the sensor.

The depth sensor consists of an infrared emitter and an infrared camera. The infrared emitter emits an array of infrared rays in such a way that the environment is filled by these rays. The moment a ray encounters an object, it is reflected, and thus captured by the IR camera. With this, it is possible to calculate the distance of the object to Kinect, and, from there, to be able to map the environment in three dimensions. The sensor calculates the time that elapses between the time the beam was emitted and the time it was captured by the camera, so that it can approximate the distance of the object. With this data we can obtain the third dimension of the objects in the image, improving the performance of the recognition of human movements and actions. In Figure 3 we can see a Kinect sensor.

Figure 3: Kinect Sensor.

4.3 Mobile Robot

The mobile robot, which received the gesture recognition system, consists of a robot driven by wheels. It features two front wheels that perform the traction to exert the movement of the robot and a third independent rear wheel with the function of giving balance and stability to the robot besides aiding it in lateral movements. Two servo motors are used (i.e. one for each front wheel), which have built-in encoders of the EMG49 model. The control of the robot is performed by the controller module Arduino Mega 2560 that uses the ATMega 2560 controller from Atmel. Figure 4 shows us a picture of the Mobile Robot.

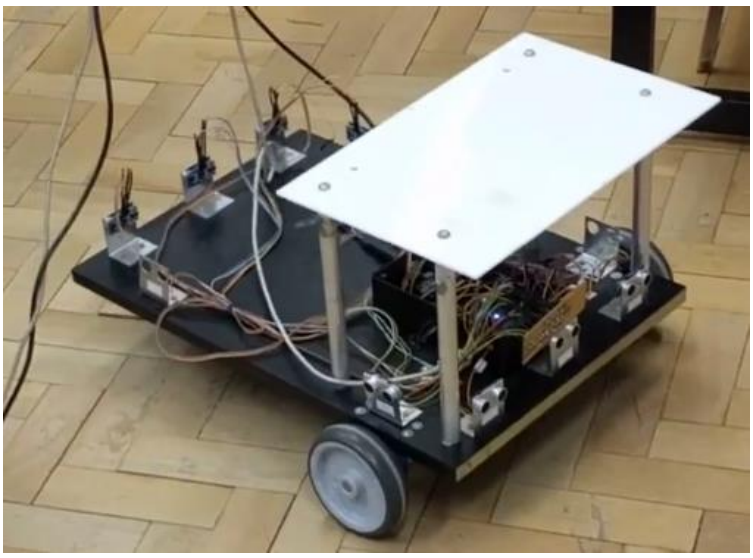


Figure 4: Mobile Robot.

4.4 Software Architecture

The proposed architecture was implemented in an Intel Core i7 3.06 GHz 4 Cores/16GB- DDR4/Ubuntu14.04-x64, and it is carried out by using python 3.6 to program in the TensorFlow framework. The joints positions were obtained using a Kinect sensor (first version). We use the Kinect SDK framework together with the PyKinect library for data collection and storage in the Python application. We send commands for the mobile robot using serial communication with the Pyserial library in python 2.7.

TensorFlow [27] is a platform for research and deployment of machine learning systems in many areas, such as computer vision and robotics. Its computational model is based on graphs of data flow with changeable state. It is widely used in research and is effective in applications involving machine learning.

The Kinect Software Development Kit (SDK) is a development framework that has come to assist developers in using Kinect, bringing integrated methods and functions not previously seen. This SDK was developed by Microsoft for Windows with the goal of enhancing the effects of Kinect with its automatic detection of joints, hands, people and objects. This has transformed human-computer interaction into a number of areas, such as education, the medical field, and transportation.

5 General System Architecture

The action detection method proposed in this work relies on a deep CNN architecture to classify an image where each row represents a frame and each column a specific joint. In order to implement our approach, it is necessary to pre-process the data by fixing its size (i.e. which is necessary for the CNN input).

The general system architecture is shown in Figure 5, and it could be summarizes in four main steps, in the first step, a Kinect sensor captures images of a person that is executing a gesture; in the second step, this image feeds a convolutional neural network, the neural network has been previously trained with data generated from the joints of the MSRC-12 data set, the convolutional network is written in python using the TensorFlow library, in the third step the network recognizes the gesture using the data collected by us. In the third step, we add a mobile robot to the system, the robot is controlled according to the results of the classification gesture algorithm. we were able to use data collected by us, also, each gesture is related to one robot movement, such as robot go forward, robot go backwards, etc. And finally, in the fourth step, we add a mobile robot to the system to control it according to the classified gesture. we were able to use data collected by us to the system to obtain tests with people in a controlled environment. We can see, in the Figure 5, the flow of the system.

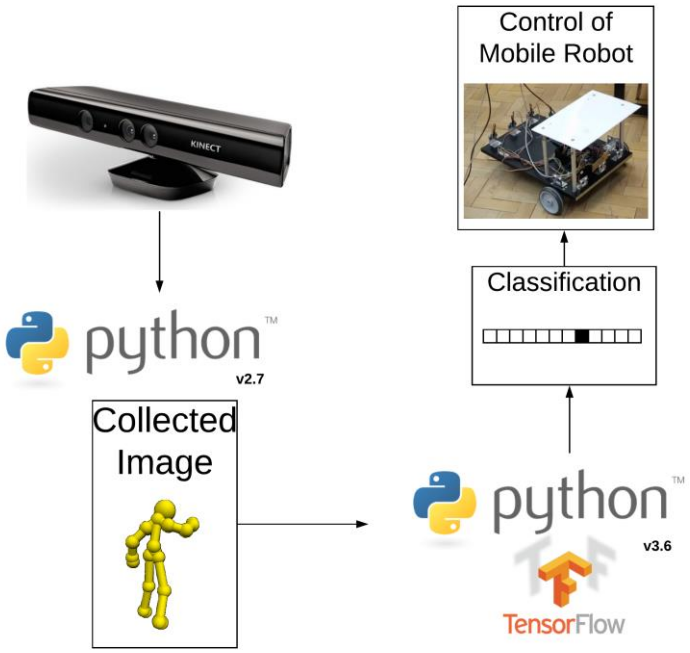


Figure 5: Flow of the gesture recognition system.

A gesture can have a different number of frames because it could be seen as an aperiodic signal, so in order to make that all the gestures could have a fixed size we used the FastDTW algorithm [2], so we created a fixed size image for the neural network input of 667x80 resolution, where 667 correspond to the number of frames and 80 represents that number of frames for every frame. The number of frames of 80 for our 20 joints is calculated using the 3D Cartesian coordinates of the 20 joints that gives 60 plus a separation of 20 between each joint that is added, so we have 60. Fig. 6 shows an example of the matrix that we created for every gesture that we used as an input to the convolutional network.

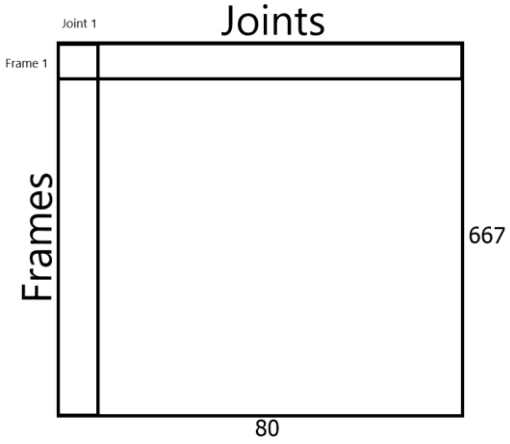


Figure 6: Input to the convolutional network.

6 Convolutional Neural Network Strategies: Combined and Individual Training

We tested two strategies for gesture recognition. In the first one, combined training, all the gestures were trained with one neural network. Then, on the second strategy, called individual training, we trained 12 neural networks, one for each gesture.

The neural network has six hidden layers, three convolutional layers with 3x3 kernel with ReLu activation functions, two pooling layers with 3x3 kernel and three strides, and one pooling layer with 2x2 kernel and two strides. In the end, it has a dense layer with 9472 units and a dropout operation with a dropout rate of 0.4 to prevent overfitting, as we can see in the Figure 7.

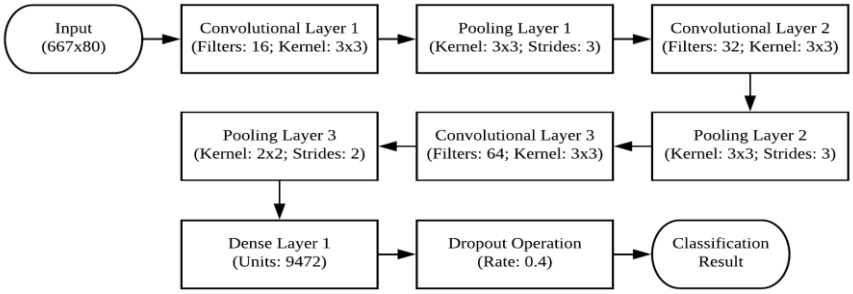


Figure 7: CNN Structure.

7 Results

In this section, we present the results regarding the experiments of training the model with the 12 gestures and the experiments with the model using the individual and combined training of the gestures in the MSRC-12 data set. In addition, we present results from data collected experimentally by us using a Kinect sensor applying also both strategies, and the results of using all the proposed architecture acquire a person gestures, classifying it and controlling a mobile robot all at the same time.

7.1 Results in the MSRC-12 data set

In a fairly large dataset, the number of frames for each action is unlikely to be repeated even when the gesture is executed by the same person, we can see the variation of that number. Using traditional convolutional networks in images we expect them all to have a fixed size and if this does not happen, we can resize them without major losses.

These actions can have a variable number of frames and that could make it hard to train the data in a fixed neural

network with a fixed-size input. Considering this problem, as pre-processing, it was necessary to use an algorithm to have all gestures with a fixed size of frames, therefore, we use the FastDTW [2] algorithm.

So, the Microsoft Research Cambridge-12 database (MSRC-12) was pre-processed using the FastDTW algorithm [3] to create images of 667x80 where 667 is the number of frames normalized by the algorithm of dynamic time warping and 80 represents the number of variables per frame. The dataset used has the positions collected from the 20 joints where each joint has three coordinates (x, y z), that is, 20 x 3 = 60 variables. Also, a separation between each joint is added, thus, we have 60 + 20 = 80 variables. So, we've shaped the network to have a fixed input of that size.

we present the results regarding the use of the MSRC-12 data set in training and validation using the proposed CNN model. We will also present the results regarding the training of the model trained with the 12 gestures in a convolutional network and the results of the individual training of the actions (12 trainings).

7.1.1 Combined Training

The dataset used has 6244 gesture samples, which we can use for training and validation. In this training, after the creation of the images in the pre-processing, we separated 33.33% of the gestures for validation and left 66.66% of the data for the training. We obtained a total of 4162 images of 667x80 for training and 2082 images of the same size for evaluation. Considering that the number of samples is relatively large for gesture recognition and with the number of 12 actions for recognition, a CNN model takes several steps to converge and even more with the use of the dropout operation. Thus, the training was performed with small batches of 10 samples and 60000 steps obtaining an accuracy of 86,67% for the validation data. Figure 8 shows the accuracy of the combined training.

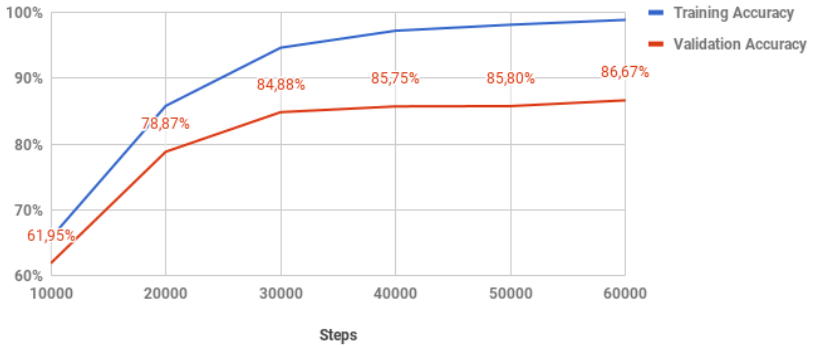


Figure 8: Combined training accuracy.

7.1.2 Individual Training

In the individual training, we used the same model for the individual training of each of the 12 gestures, that is, we trained 12 separate networks. The training was done using the number of samples of a gesture plus the same number of samples from other randomly selected gestures. Thus, each individual network was trained with approximately 1000 samples. Figure 9 shows the validation and training accuracy of one of the individual training networks. Besides, we can notice in the Figure 10 the accuracy of each individual training, with 30000 steps, together with the general accuracy obtained by the weighting of the individual accuracies.

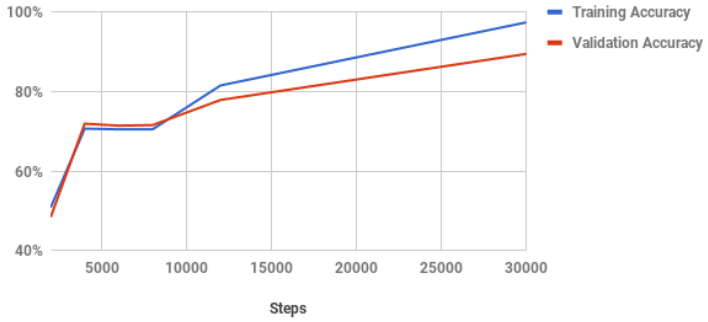


Figure 9: Individual Accuracy of Wind it up' gesture.

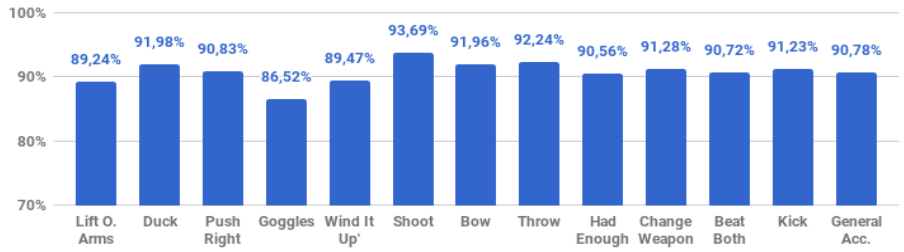


Figure 10: Individual training accuracy of all the Gestures of the evaluation set.

7.2 Experimental Results with Gestures Captured from a Person with the Kinect

After the training and verification of the neural network we passed the execution and classification using real data collected from Microsoft Kinect (v1) with a person, and, we have also used the two strategies, named combined and individual training, for classification. As previously explained, it was necessary to make the communication between the process that collects the data (Python 2.7) and the classification process (Python 3.6), which made classification possible. We used 20 samples of each movement and in each training format of the network, that is, 480 samples of gestures were collected by us.

7.2.1.1 Classification using combined training

Figure 11 shows the results of the accuracy of the system obtained after the training processes and using data captured from the Kinect with a person realizing a gesture. We can observe in the figure that in some movements, such as 'duck' and 'throw', the neural network performs very well, but in others such as 'goggles' and 'wind it up' we find that the performance was well below expectations. We believe this is due to the fact that it is difficult to execute some movements and they are easily confused both by the trained network and by humans, we take the example of the 'googles' movement that the trained network has confused several times with the 'had enough' movement.

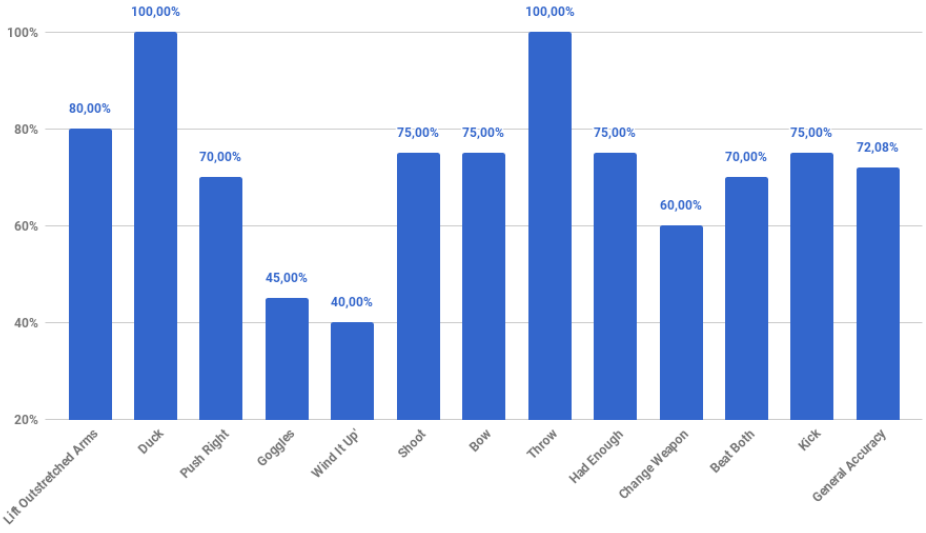


Figure 11: Combined training accuracy on evaluation set using data collected with the Kinect sensor and a person.

7.2.1.2 Classification using individual training

The experiments were repeated using the strategy of the individual training, but this a person will execute a determined movement instead of using the information of the MSRC-12 data set. Figure 12 shows the results accuracy of the individual training accuracy of the network obtained with this strategy, we can observe in the figure, that the accuracy was satisfactory and closer to the performance seen using the evaluation set of MSRC-12.

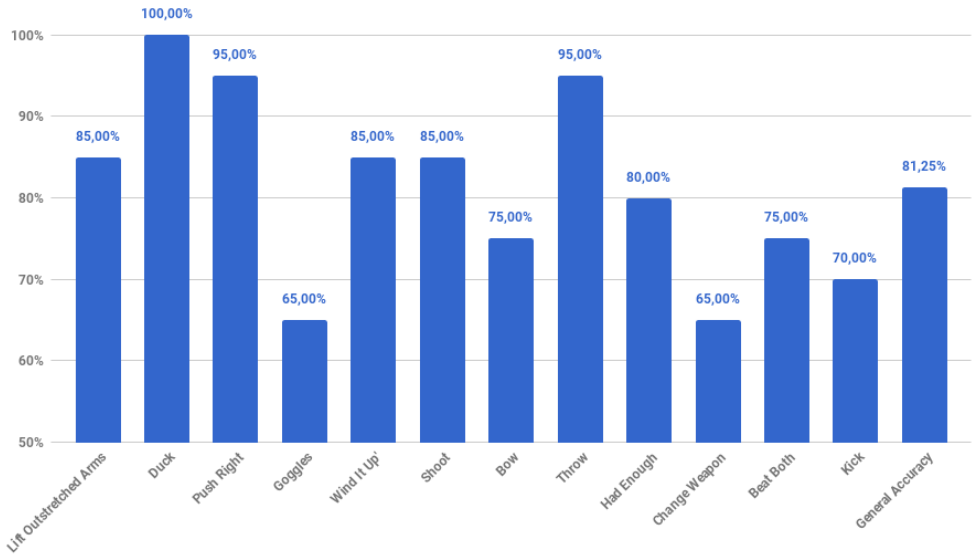


Figure 12: Individual training accuracy on evaluation set using data collected with the Kinect sensor and a person.

7.3 Results Controlling the Mobile Robot

Finally, after the two set of experiments using both training strategies in each of them, we decided that besides including a person it could be possible to include a robot in the control loop, closing the architecture proposed that is shown in Figure 5. The results pointed out that controlling the mobile robot did not cause problems for the execution of the system. Thus, we obtained the same results using the neural network for the gesture classification. Every gesture is related to a predefined robot movement. So, we reached the goal of using the system to control a mobile robot. We can see, in the Figure 13 a sequence of frames from the movement throw, followed for the sequence of movements deployed for the mobile robot in 4 different moments of the experiments.

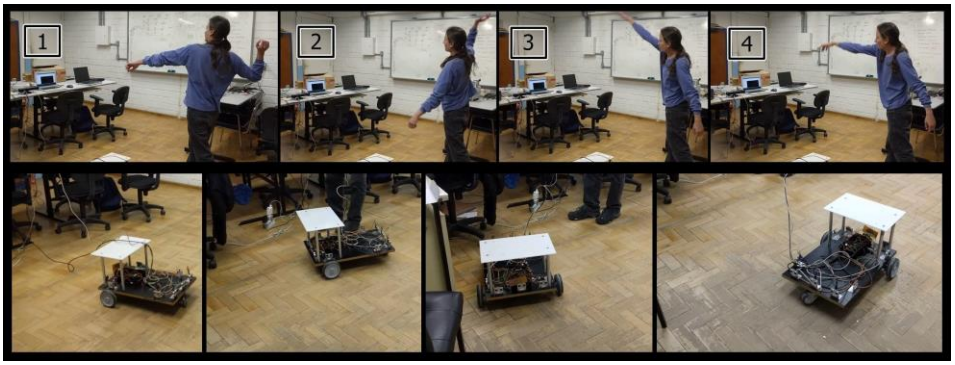


Figure 13: Sequence frames from the movement throw.

After the classification of the movement which was made, according to the gesture predicted by the convolutional network, a command is sent to control the robot. For testing, we used four gestures trained to command the robot: Lift arms, duck, throw and shoot. Any other gesture (provided by the network) is sent as the stop motion command. From these commands, the robot makes one of four simple coded movements: Square anti-clockwise (lift arms); Square clockwise (duck); Counter clockwise circle (shoot); Clockwise circle (throw). Using the robot odometry, it is possible to obtain the position and orientation of the robot. Table 1 shows the cartesian positions captured from square movement counter clockwise developed by the robot. Also, Figure 14 shows the trajectory described by the robot following the predefined trajectory. The performance of the whole system can be seen in this section. Fig. 13 shows the sequence of movements of the robot for the throw gesture, in this case the robot has to make a square following a counter clockwise rotation and the positions of the robot in the four-square corners measured with the odometry of the robot are shown in Table 1, the coordinates captured demonstrates that the robot executed the predefined movement associated to the gesture throw.

Table 1: Positions captured from square movement counter clockwise.

	x coordinate	y coordinate
1	20,24885	0,00000
2	20,22809	-0,21179
3	20,42756	-0,22335
4	20,43920	-0,04351

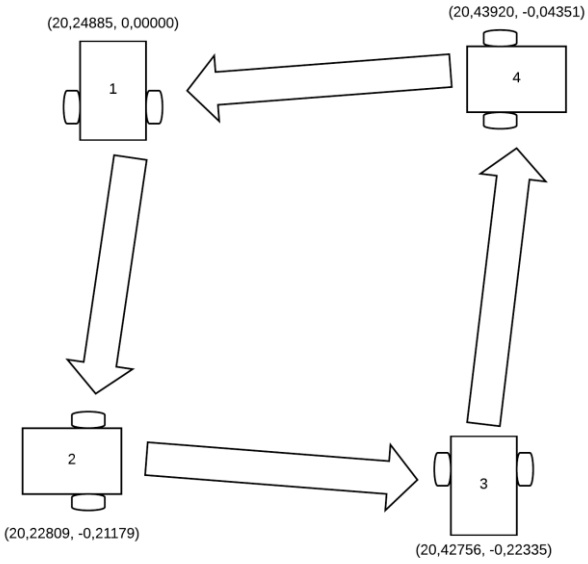


Figure 14: Square movement counter clockwise.

7.4 Results Analysis

In relation with the experiments realizes in the MSRC-12 data set, we can see that individual training achieved an accuracy of 4 percent higher than the combined training. This difference could be explained mainly by the similarity of some images (actions) which makes the distinction in a single network more difficult. On the other

hand, in small networks, this fact could have a minor influence because each network is responsible for recognizing only one gesture. Training the individual networks brings a gain in accuracy, as discussed earlier, but training and execute time is greatly increased and could compromise their use in applications where is required a fast response, for example, a real time application.

Neural network structures based on CNN do not recognize rotated images (if there is no image with the same rotation in the training set). Wu [28] demonstrates that convolutional networks only become efficient and capable in recognizing rotated images if rotation layers as rotated-pooling or flip-rotated-pooling convolution are added to the convolutional model. This CNN characteristic could be a problem for many applications, such as image recognition in general. On the other hand, in our case, this characteristic can be seen as beneficial, because with the movement of articulations other than the originals there is a significant change of the gesture and cannot be confused with the trained and correct action.

We would like to drive the attention now to the classification experiments using data collected by us with persons, we have seen a good performance in some gestures, but below-expected performance in others. We believe that it is a problem caused because some gestures have some similar characteristics. The same results were obtained using the neural network for the gesture classification controlling the robot. The robot was controlled to make simple movements, which, as seen earlier, brought satisfactory results.

Also, we would like to remark that our application shows that it is possible to integrate machine learning methods with robotics as can be explored in other works [29], [30].

8 Conclusion

CNN models are very popular in computer vision problems, but less exploited in other areas. The answer obtained in this paper by a model of a convolutional network demonstrates that it is possible and efficient its use in the recognition of gestures from joints obtained from the Kinect sensor. Furthermore, we proposed and experimented with strategies for the training of the CNN, named the combining and individual training, It was possible to observe that training of 12 smaller networks increased the training time substantially, but it is obtained a higher accuracy that makes this approach more useful and avoid a bottleneck with the training time, it is also possible to parallelize the individual trainings since they are independent of each other. So, these strategies, combined and individual training, is another contribution of the paper. In addition, we collect data from a Kinect sensor to use the system and, thus, get results closer to reality. The use of the Kinect sensor was satisfactory, although, the correctness rate for some gestures did not follow the results seen in the database. The use of the system to control a mobile robot was possible and its performance satisfactory. Some future works could focus on the use of a Recurrent Neural Network instead of a CNN.

9 References

[1] Mo, L., Li, F., Zhu, Y., Huang, A.: Human physical activity recognition based on computer vision with deep learning model. IEEE Int. Instr. and Meas. Technology Conf. Proceedings, Taipei, pp. 1-6 (2016).

[2] MSRC-12 dataset: https://www.microsoft.com/en-us/download/details.aspx?id=52283, last accessed 2018/08/21.

[3] Salvador, S., Chan, P.: FastDTW: Toward accurate dynamic time warping in linear time and space. Intelligent Data Analysis 11.5, pp. 561-580 (2007).

[4] Hou, Y., Li, Z., Wang, P., Li, W.: Skeleton optical spectra based action recognition using convolutional neural networks. IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 3, pp. 807-811 (2018).

[5] Jiang, X., Zhong, F., Peng, Q., Qin, X.: Online robust action recognition based on a hierarchical model. 30: 1021. https://doi.org/10.1007/s00371-014-0923-8 (2014).

[6] Ke, Q., An, S., Bennamoun, M., Sohel, F., Boussaid, F.: SkeletonNet: Mining Deep Part Features for 3-D Action Recognition. In: IEEE Signal Processing Letters, vol. 24, no. 6, pp. 731-735 (2017).

[7] Sharaf, A., Torki, M., Hussein, M. E., El-Saban, M.: Real-time multi-scale action detection from 3D skeleton data. IEEE Winter Conf. on Applications of Computer Vision, Waikoloa, HI, pp. 998-1005 (2015).

[8] Nguyen, D., Le, H.: Kinect Gesture Recognition: SVM vs. RVM. Seventh International Conference on Knowledge and Systems Engineering (KSE), Ho Chi Minh City, pp. 395-400 (2015).

[9] Pan, H., Li, J.: Online human action recognition based on improved dynamic time warping. IEEE International Conference on Big Data Analysis (ICBDA), Hangzhou, pp. 1-5 (2016).

[10] Xia, L., Chen, C. C., Aggarwal, J. K.: View invariant human action recognition using histograms of 3D joints. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Providence, RI, pp. 20-27 (2012).
[11] Piyathilaka, L., Kodagoda, S.: Gaussian mixture based HMM for human daily activity recognition using 3D skeleton features. IEEE 8th Conference on Industrial Electronics and Applications (ICIEA), Melbourne, VIC, pp. 567-572 (2013).
[12] Althloothi, S., M. Mahoor, H., Zhang, X., Voyles, R. M.: Human activity recognition using multi-features and multiple kernel learning. Pattern Recognition, V. 47, Issue 5, pp. 1800-1812 (2014).
[13] Du, Y., Fu, Y., Wang, L.: Representation Learning of Temporal Dynamics for Skeleton-Based Action Recognition. In: IEEE Transactions on Image Processing, vol. 25, no. 7, pp. 3010-3022 (2016).
[14] Oyedotun, O.K., Khashman, A.: Deep learning in vision-based static hand gesture recognition. Neural Comput and Applic, pp. 28 (2017).
[15] Borja, J. A. T., Alzate, E. B., Lizarazo, D. L. M.: Motion control of a mobile robot using kinect sensor. IEEE 3rd Colombian Conference on Automatic Control (CCAC), Cartagena, pp. 1-6 (2017).
[16] Wang, Y., Song, G., Qiao, G., Zhang, Y., Zhang, J., Wang, W.: Wheeled robot control based on gesture recognition using the Kinect sensor. IEEE International Conference on Robotics and Biomimetics (ROBIO), Shenzhen, pp. 378-383 (2013).
[17] Zhao, X., Naguib, A. M., Lee, S.: Kinect based calling gesture recognition for taking order service of elderly care robot. The 23rd IEEE International Symposium on Robot and Human Interactive Communication, Edinburgh, pp. 525-530 (2014).
[18] Limin, M., Peiyi, Z.: The medical service robot interaction based on kinect. IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), Srivilliputhur, pp. 1-7 (2017).
[19] Fadli, H., Machbub, C., Hidayat, E.: Human gesture imitation on NAO humanoid robot using kinect based on inverse kinematics method. International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation (ICAMIMIA), Surabaya, pp. 116-120 (2017).
[20] Kundu, A. S., Mazumder, O., Chattaraj, R., Bhaumik, S.: Door negotiation of a omni robot platform using depth map based navigation in dynamic environment. Seventh International Conference on Contemporary Computing (IC3), Noida, pp. 176-181 (2014).
[21] Abdelkrim, N., Issam, K., Lyes, K., Khaoula, C.: Fuzzy logic controllers for Mobile robot navigation in unknown environment using Kinect sensor. IWSSIP Proceedings, Dubrovnik, pp. 75-78 (2014).
[22] Kameyama, N., Hidaka, K.: A sensor-based exploration algorithm for autonomous map generation on mobile robot using kinect. 11th Asian Control Conference (ASCC), Gold Coast, QLD, pp. 459-464 (2017)
[23] Bellarbi, A., Kahlouche, S., Achour, N., Ouadah, N.: A social planning and navigation for tour-guide robot in human environment. 8th International Conference on Modelling, Identification and Control (ICMIC), Algiers, pp. 622-627 (2016)..
[24] Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., Lew, M.l S.: Deep learning for visual understanding: A review. Neurocomputing. Recent Developments on Deep Big Vision, pp. 187:27 – 48 (2016).
[25] Zeiler, M.: Hierarchical Convolutional Deep Learning in Computer Vision. Ph.D. thesis, New York University (2014).
[26] Chu, S., Keogh, E., Hart, D., Pazzani, M.: Iterative Deepening Dynamic Time Warping for Time Series. Proceedings SIAM International Conference on Data Mining, pp. 195-212 (2002).
[27] Tensorflow: Tensorflow Framework documentation. https://www.tensorflow.org, last accessed 2018/10/14.
[28] Wu, F., Hu, P., Kong, D.: Flip-Rotate-Pooling Convolution and Split Dropout on Convolution Neural Networks for Image Classification. arXiv preprint arXiv:1507.08754v1 (2015).
[29] Pinpin, K., Gamarra, D.F.T., Johansson, R., Laschi, C., Dario, P.: Utilizing Gaze Behavior for Inferring Task Transitions Using Abstract Hidden Markov Models. Inteligencia Artificial, v. 19, pp. 1-16, (2016).
[30] Silva, R.M., Cuadros, M.A.S.L., Gamarra, D.F.T.: Comparison of a Backsteping and a Fuzzy Controller for Tracking a Trajectory with a Mobile Robot. 18th International Conference on Intelligent Systems Design and Applications (ISDA), (2018).