Action recognition based on 2D skeletons extracted from RGB videos

Sophie Aubry, *Sohaib* Laraba[*], *Joëlle* Tilmanne, and *Thierry* Dutoit

University of Mons, TCTS Lab, 31, Boulevard Dolez B-7000 Mons, Belgium

Abstract. In this paper a methodology to recognize actions based on RGB videos is proposed which takes advantages of the recent breakthrough made in deep learning. Following the development of Convolutional Neural Networks (CNNs), research was conducted on the transformation of skeletal motion data into 2D images. In this work, a solution is proposed requiring only the use of RGB videos instead of RGB-D videos. This work is based on multiple works studying the conversion of RGB-D data into 2D images. From a video stream (RGB images), a two-dimension skeleton of 18 joints for each detected body is extracted with a DNN-based human pose estimator called OpenPose. The skeleton data are encoded into Red, Green and Blue channels of images. Different ways of encoding motion data into images were studied. We successfully use state-of-the-art deep neural networks designed for image classification to recognize actions. Based on a study of the related works, we chose to use image classification models: SqueezeNet, AlexNet, DenseNet, ResNet, Inception, VGG and retrained them to perform action recognition. For all the test the NTU RGB+D database is used. The highest accuracy is obtained with ResNet: 83.317% cross-subject and 88.780% cross-view which outperforms most of state-of-the-art results.

1 Introduction

The task of recognition is to name actions and describe them from a sequence of movements. The study of human action is linked to many fields such as computer science, psychology, healthcare. Action recognition can be used in many applications such as video surveillance, human-machine interaction, user interface design, gaming, entertainment, robotics, web-video description, medical diagnosis, sports analysis and many others. Action recognition is a major challenge because of the diversity of movements, the complexity of motion capture and the difficulty of creating a realistic and relevant database. An action, such as drinking, can be done in different ways, depending on the person acting, the context, the environment, the style of the movement and many other parameters. Each action can be done in so many ways that it is difficult to define the features describing the action. Another challenge is how to capture and represent movement. Different representations exist and depending on the final application, many motion capture systems are available. To obtain good results in the recognition task, a large amount of data is necessary and specially to

[*] Corresponding author: sohaib.laraba@umons.ac.be

train a neural network. All this data is gathered in a database. For action recognition, the creation of a database and manual labelling of all movement sequences one by one takes a lot of time. In addition, the database should be as representative as possible of the reality and conditions under which the recognition algorithm will be used.

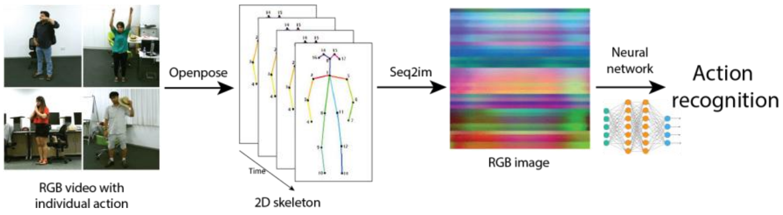

Fig. 1. Methodology used to recognize action: input data are RGB video containing individual actions (from NTU RGB+D dataset), the skeleton is extracted with Openpose then the sequence of motion is converted into a RGB image before going into the neural network which classifies the action.

We present a new methodology to process RGB videos to perform action recognition (Fig. 1). The first step to recognize human activity is to extract the motion from the video. As a result, the essence of the motion is extracted, namely, the skeleton of people. Skeleton extraction is done using OpenPose [1], a DNN-based detection system that extracts a 2D skeleton of 18 joints for each detected body. Second, motion sequences are converted to RGB images. The classification of images and videos has been very successful with the development of deep learning. We exploit these advantages by transforming the skeleton-based action recognition task into an image classification task. The motion parameters are encoded in the three R, G, B channels and an action sequence becomes an RGB image. Finally, image classification neural networks can be retrained to recognize actions.

2 State of the Art

Image and video classification have known a great success in the last few years, and especially with the development of Deep Learning. Some researches tried to exploit these advantages by transforming the skeleton-based action recognition task into an image classification task, and this by transforming 3D skeleton sequences into images. Image classification models can be then used on this data [2]. The motion parameters are encoded into the three R, G, B channels and action sequences become an RGB image. Then the image classification neural network can be retrained to recognize actions.

Laraba et al. [2] encode the (X, Y, Z) position of each joint into R, G, B channels. They obtain with a CNN an accuracy of 74.27% for the cross-subject and 75.74% for the cross-view action recognition using 3D skeleton data recorded by the Kinect from NTU RGB+D dataset [3]. In the same idea, Du et al. [4] obtain an accuracy of 100% with the Berkeley MHAD dataset [5].

In [6], Ding et al. investigated different skeleton features to encode them into RGB images. They test five different features: joint-joint distances, joint-joint vectors, joint-joint orientations, joint-line distances and line-line angles. They encode these features into five RGB images before using them to train five CNN. Then, all the output scores are fused to return a final score of the classification. With this method they achieve an accuracy of 82.31% with the NTU RGB+D dataset [3].

In [7], Ke et al. suggest a new representation of skeleton data. They encode the 3D coordinates (X,Y,Z) into a clip of grey images containing spatio-temporal information. They obtain an accuracy of 84,83% cross-view and 79.57% cross-subject with the NTU RGB+D database [3], 93.57% with the SBU Kinect interaction database [8] and 88.30% with the CMU dataset [9].

Wang et al. [10] propose a joint trajectory map in which 3D skeleton data are encoded. These maps are then classified by a CNN. Their joint trajectory map encodes the 3D coordinate of the joint, the motion direction, the body parts and the magnitude of the motion. They obtain an accuracy of 81.08% cross-view and 76.32% cross-subject with the NTU RGB+D database [3], 94.86% with the MSRC-12 Kinect Gesture Dataset [11] and 96.02% with the G3D DATASET [12].

Li et al. [13] suggest to encode 3D relative coordinates and compute the difference with 3 reference joints (hip centre, right shoulder and left shoulder). The three generated sets of vectors containing relative coordinates are encoded into an RGB image. They train a CNN with these images and get an accuracy of 75.2% in cross-subject and 82.1% in cross-view with the NTU RGB+D database [3].

In [14], Li et al. propose to use skeleton data and divide them into five body parts, the 3D coordinates (X, Y, Z) of joints of each body part is concatenated in a vector and encoded into a RGB image. The RGB images feed the CNN and obtain with the NTU RGB+D dataset [3] an accuracy of 84.6% cross-subject and 90.9% cross-view.

Li et al. in [15] rearrange and select the important skeleton joints automatically. The order of joints during encoding influences the accuracy. The module selects the best order and the most important joints before encoding them into an image and feed the CNN. They obtain an accuracy of 83.2% cross-subject and 89.3% cross-view.

3 Methodology

The proposed methodology is based on RGB videos. Motion information are extracted with OpenPose. These skeleton motion sequences are then converted into images. In this section, OpenPose is presented then the conversion of the motion sequence into RGB image is explained.

3.1 Machine Specification

For this work, all the tests on neural networks for classification algorithms are performed on a computer, equipped with a MSI GeForce_GTX 1080 TI GAMING X 11G GPU and 32 GB (2x16GB DDR4 3000) memory. The computer is also equipped with a Intel Core Skylake-X I7-7800x 3.5Ghz 25Mb LGA 2066 BOX CPU.

3.2 Motion Extraction with OpenPose

OpenPose is able to detect the 2D poses of several people in an image. Detection is done by a bottom-up approach: body parts are detected and then associated with individuals on the image. Cao et al. [1] propose a two-flow CNN architecture: the first branch is dedicated to limb detection and the second to the affinity field of the part that gives the degree of association between body parts to determine which are associated with which body (Fig. 2). This step is done iteratively: each step takes as input the detection confidence cards and the affinity field of the part (the two output branches) of the previous step and the original image. OpenPose extracts the position of 18 joints for each detected body and returns for each joint the coordinates (X, Y) and the confidence score C relative to the detection of each joint.

In this work, we focus on the recognition of individual actions, only the first 49 actions are retained from the NTU RGB+D database. From all RGB videos in the database, the actor's skeleton is extracted with OpenPose. Even if OpenPose does not reach the same performances as 3D motion capture systems, OpenPose requires only RGB camera and extract 2D skeletons from a video. It can be used in indoor and outdoor environments.

OpenPose allows partial skeleton recognition. Some disadvantages of OpenPose are that it returns no information about the depth and it is based on DNN which requires a high-end computer.

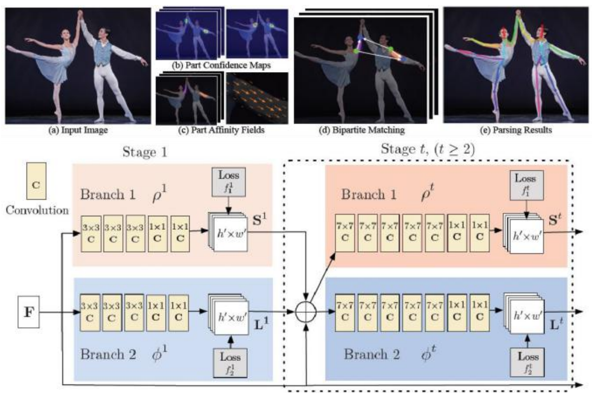

Fig. 2. OpenPose methodology for the 2D pose detection of several people in an image and neural network architecture for detection and association of body parts [1].

On some videos, OpenPose extracts the skeleton of people in the background unrelated to the action recognition task. The extracted data have been cleaned and only the first skeleton of 18 joints remains.

3.3 Data Conversion

Neural networks in the field of image classification have evolved rapidly and many models have emerged. With the large number of models in this field and the good performance of these models, a representation of skeletal motion sequences in images has been proposed by Laraba et al. [2]. The conversion of the motion sequence into an RGB image is done in three steps: extraction of the skeleton data in the matrices (X, Y, Z), normalization between 0 and 255 and mapping with the R, G, B channels. With this transformation, the motion sequence is reduced to an image.

In this work, the motion is recorded through traditional RGB camera. From these videos, the motion is extracted thanks to OpenPose which returns the (X, Y) positions of the joints and the confidence C of the detection of joints for each frame. (X, Y) and C are extracted into three different matrices. For each one, a row corresponds to the position of a joint along the time and a column to the positions of all the joints at one frame. These three matrices are then normalized between 0 and 255 before being mapped into RGB images. With this transformation each sequence of video representing an action is converted into a single RGB image (Fig. 3).

4 Data representation

In this section, an evaluation of the movement parameters influencing the effectiveness of the classification will be conducted. Before feeding them to the neural network, the motion capture data will be transformed into an image. For each joint, three channels are available (RGB). The experimentation consists in finding the best data representation and the best kind of encoding to have the highest accuracy. To realize this experimentation, two models of image classification neural network have been tested: SqueezeNet [16] and DenseNet [17]. SqueezeNet is very small and has very few parameters, which makes it quick to be trained. DenseNet is a more complex neural network, with a high number of parameters to optimize. Due to its complexity, the training time is higher, but the accuracy is improved compared to SqueezeNet.

All the following tests (summarized in Table 1) have been realized with a deep retraining. We use models pretrained on the ImageNet dataset, and we retrain the whole network. We prefer to use a deep retraining instead of a shallow retraining because our data are very different from the ImageNet dataset.

The best data representation will be then used with more complex neural network architectures. All the data representations have been tested to do cross-subject and cross-view recognition.

4.1 Test 1: Position (X, Y) and Confidence

For the first experimentation, for each motion sequence, (X, Y) position and the confidence score C given by OpenPose for each joint are encoded respectively in the R, G and B channels of the image (RGB) (Fig. 3).

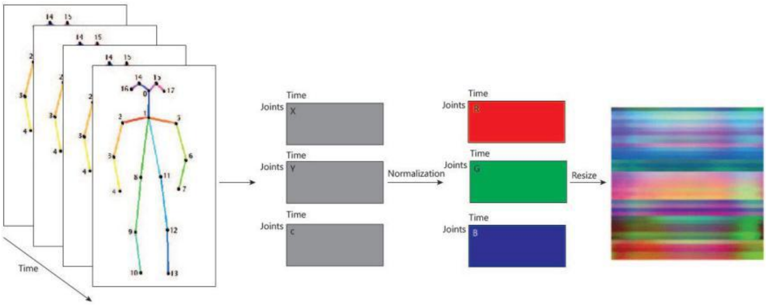

Fig. 3. (X, Y) positions and confidence C of each joint are encoded in R, G and B channels of an image.

The accuracy obtained with SqueezeNet is equal to 74.25% and it is equal to 82.71% with DenseNet (Tables 2 and 3).

4.2 Test 2: Position (X, Y) and mean between (X, Y)

For the second test, we encode the mean value between (X, Y) as input information about the depth. The size of the body in the video gives information about the closeness of the actor. If the actor is far from the camera, he looks small on the image. When he comes closer to the camera, he is larger on the image. Some information about the depth is hence contained into the (X, Y) position of each joint. We propose to encode the (X, Y) positions and the mean between X and Y in the R, G and B channel of an image (Fig. 4).

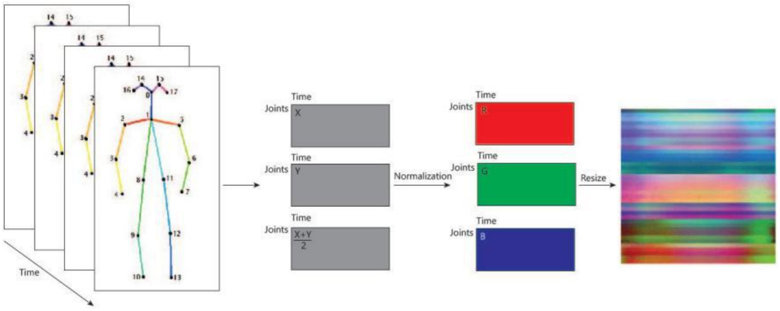

Fig. 4. (X, Y) position and the mean of (X, Y) of each joint are encoded as an image.

The accuracy obtained with SqueezeNet is equal to 71,43% and to 80,33% with DenseNet (tables 2 and 3). These two first tests show that putting the confidence C in the encoding is more relevant than setting the mean value between (X, Y) coordinates.

4.3 Test 3 and Test 4: 14 Joints

For the third and fourth experiments, the two data encoding presented above are tested with only 14 joints instead of 18 (Fig. 5). Some articulation positions of the skeletal extraction performed with OpenPose are missing. In particular, the joints of the eyes and ears are often absent. For some gestures, they are missing in 50% of the frames. The actions we want to recognize do not involve the eyes or ears. The neural network was retrained by keeping only the other 14 nodes.

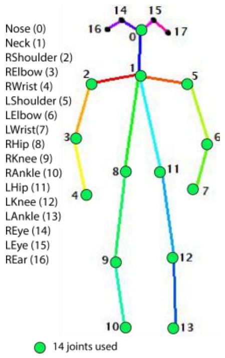

Fig. 5. OpenPose 2D skeleton [18]

The tests with 14 joints are realized one time with the confidence C and a second time with the mean value between (X, Y) coordinates as the third channel B. The results of the tests are in Tables 2 and 3. Using the confidence C, the test 1 and test 3 must be compared together. The removal of the ears and eyes nodes shows an improvement with SqueezeNet [16] and a small regression with DenseNet [17]. Test 2 and test 4, using the mean value between (X, Y) coordinates, show an improvement of the accuracy with the two models when we use 14 joints instead of 18. It seems using 14 joints instead of 18 joints gives better accuracy.

4.4 Test 5 and Test 6: Change Joints Order

For these tests, the (X, Y) positions of the 14 joints are encoded in different orders. The understanding of the motion by the neural network depends on the data representation and the order of the joints in the encoder gives different accuracy results. The first order tried in tests 3 and 4 is the following: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13. The second order tried in tests 5 and 6 is the following: 0, 4, 3, 2, 1, 5, 6, 7, 10, 9, 8, 11, 12, 13 (Fig. 5). This order associates the two arms and the two legs.

When we compare the results of the tests 3 and 4 with tests 5 and 6, we can conclude that the accuracy is influenced by the order of the joints and the order 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 gives a better accuracy (Tables 2 and 3).

4.5 Conclusion: the best data representation

To find the best data representation, the different data representations have been tested with two models: SqueezeNet [16] and DenseNet [17]. These tests show that:

- the more relevant third coordinate is the confidence C returned by OpenPose and it is obvious the (X, Y) coordinates are neccessary.

●the suppression of the most missing joints (eyes and ears) is relevant and improves the accuracy with Squeezenet.

●the order of the joints is important and the best order found is: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.

Table 1. Parameters used in the tests to evaluate the different data representations.

test	RGB conversion	number of nodes	nodes order
1	(X,Y),c	18	order 1
2	(X,Y),mean	18	order 1
3	(X,Y),c	14	order 2
4	(X,Y),mean	14	order 2
5	(X,Y),c	14	order 3
6	(X,Y),mean	14	order 3

legend
order 1 = (14,15,16,17,0,1,2,3,4,5,6,7,8,9,10,11,12,13)
order 2 = (0,1,2,3,4,5,6,7,8,9,10,11,12,13)
order 3 = (0,4,3,2,1,5,6,7,10,9,8,11,12,13)

Table 2. Results of the tests to evaluate the different data representations with SqueezeNet [16] in cross-subject.

	Memory size	parameters optimized
SqueezeNet	3	747 633

Results	Accuracy (in%)	Training time (min)	Evaluation time (s)
test1	74.253%	22.80	29.2680
test2	71.439%	22.34	29.2983
test3	**74.553%**	22.50	29.9367
test4	73.160%	22.57	29.0531
test5	73.512%	22.83	29.4864
test6	72.487%	22.75	29.2175

As shown in Table 2, with the SqueezeNet model, the training time is approximately the same for all the tests and is between 22.34 and 22.83 minutes. The evaluation time is also approximately the same for all the tests and is between 29.0531 seconds and 29.9367 seconds. The highest accuracy obtained is 74.553%.

The DenseNet model is more complex. Compared to SqueezeNet, DenseNet has a higher number of parameters to optimize and training time and evaluation time are higher. The training time is between 73.16 and 74.63 minutes, which is approximately three times higher than SqueezeNet. The evaluation time is between 55.6769 and 57.2752 seconds which is approximately twice the evaluation time of SqueezeNet. The highest accuracy obtained is 82.718%.

The two models do not agree on the best data representation. According to SqueezeNet, the best data representation is obtained with the parameter configuration of the test 3 and with DenseNet, the best accuracy is obtained with parameters configuration of the test 1. The difference between these two tests is the number of joints: 14 or 18 joints.

In terms of accuracy, the best data representation is the encoding of the (X, Y) position and the confidence C of the 14 (or 18) joints in the following order: (14, 15, 16, 17), 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13. This way of encoding is used for the tests in the following section.

Table 3. Results of the tests to evaluate the different data representations with DenseNet [17] in cross-subject.

	Memory size	parameters optimized
DenseNet	51.1	12 566 065

Results	Accuracy (in%)	Training time (min)	Evaluation time (s)
test1	**82.718%**	73.39	56.0388
test2	80.338%	74.43	55.6769
test3	82.711%	74.63	57.2752
test4	80.981%	74.56	57.1248
test5	82.688%	73.16	56.1067
test6	80.735%	74.37	57.0115

5 The Image Classification Model for Action Recognition

With the best data representation chosen in the previous section, different models of image classification are compared. Then the importance of the transfer learning approach is demonstrated with different kinds of training with each model: deep, shallow or from scratch. Finally, a comparison of performances is made for the cross-subject and cross-view action recognition tasks.

5.1 Models Comparison

In this section, we compare different models of image classification in terms of accuracy, training time and evaluation time. All these models are tested with two chosen data representations (corresponding to test 1 and 3 in 1): encoding (X, Y) and C in RGB channels for 14 (or 18) nodes in the following order (14, 15, 16, 17), 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13. We evaluate the following models: SqueezeNet [16], AlexNet [19], DenseNet [17], Resnet [20, 21], Inception [22, 23, 24], VGG [25].

Three deep retraining of all the selected models have been realized with the two data representations. The best results obtained for each model are summarized in Table 4. The highest accuracy is **83.317%** and is obtained with ResNet152 with 18 nodes. It seems difficult to choose the best data representation between 18 or 14 joints. For some models (SqueezeNet, DenseNet and VGG13), suppressing the more often missing joints can improve the accuracy. With 14 nodes data representation, the highest accuracy is obtained with ResNet152 (**83.317%**). With 18 nodes, the highest accuracy is obtained with DenseNet (**82.651 %**).

The accuracy is not the only criteria to choose a model. Depending on the application, the memory required, or the training and evaluation times can be important. The models requiring the shortest training time are SqueezeNet and AlexNet. SqueezeNet requires the smallest memory size. The slowest model in terms of training time and evaluation time is ResNet152. VGG19 is the model requiring the largest memory size.

5.2 Transfer Learning: Deep, Shallow or from scratch

In this section, we evaluate the importance of the transfer learning. The different models of image classification will be retrained with a deep, shallow or from scratch training [26][27][28]. These models are originally conceived for image classification and we test their ability to perform action recognition by converting the 2D skeleton data into RGB image.

All the models have already been trained for image classification. The following models are tested: SqueezeNet [16], AlexNet [19], Inception [23], DenseNet [17], ResNet [20, 21], VGG [25].

Table 4. Results obtained by a deep retraining using 18 nodes and using 14 nodes parameters configuration in cross-subject.

Model (retrained deep)	Memory size (MB)	Parameters optimized	Accuracy for test 1(%)	Accuracy for test 3 (%)	Training time (min)	evaluation time (s)
SqueezeNet	3	747633	75.788	75.803	22.541	29.263
AlexNet	228.8	57204593	74.545	74.051	22.730	27.619
Inception v3	98.2	24481346	81.985	81.528	87.567	63.738
DenseNet169	51.1	12566065	81.940	**82.651**	74.683	55.322
ResNet34	85.4	21309809	82.591	81.356	40.366	36.053
ResNet152	233.8	58244209	**83.347**	81.693	117.074	74.771
VGG13	516.6	129151601	79.096	78.871	77.244	52.718
VGG19	559.8	139770993	78.497	78.984	109.260	66.915

The two chosen data representations are used to train the neural network models.

Transfer learning can be compared to learning a language. For someone who already knows several languages, it is easier to learn a new one compared to someone who only speaks one language. Thanks to previous knowledge, the learning process is easier. It is the same for neural networks. We start with models designed for image classification. The network is already able to classify images, and with some adaptation of the parameters it is also capable to classify motion sequence images. The three kinds of retraining are tested with different models of image classification neural networks (Table 5). As expected, the accuracy is better when a deep retraining is applied. A training from scratch gives an accuracy between 62.278% and 77.779%. It is better than the accuracy obtained with a shallow retraining which is between 26.607% and 47.025%. The shallow retraining only retrains the last layers. In this case, the difference between the features for the images in the ImageNet dataset and our images representing motion sequences are considerable. It is important to note that the parameters of the simulation are kept the same between the different kinds of retraining, especially the number of epochs which indicates the number of training iterations over the dataset. Training time is smaller for a shallow retraining because only the last layers are retrained. Since the architecture of the models is kept between the different kinds of retraining, the memory required of each model stays the same size. The evaluation time also stays the same for the different kinds of retraining.

Table 5. Results obtained by a training from scratch, deep retraining or shallow retraining and the number of the test for the data representation used (Table 1).

	From Scratch			Deep Retraining			Shallow Retraining		
Models	Accuracy (%)	Training Time (min)	Test	Accuracy (%)	Training time (min)	Test	Accuracy (%)	Training time (min)	Test
SqueezeNet	65.399	23.187	1	75.803	22.541	3	36.389	18.054	3
AlexNet	68.917	22.692	3	74.545	22.647	1	30.147	16.907	3

Inception	75.189	87.291	1	81.985	87.246	1	30.043	40.110	3
DenseNet	77.636	74.458	1	82.651	74.683	3	47.025	34.276	3
ResNet34	77.779	40.195	1	82.591	39.620	1	41.105	22.523	3
ResNet152	72.547	117.145	3	83.317	115.029	1	40.985	46.783	3
VGG13	72.854	77.287	1	79.096	76.006	1	33.366	31.956	3
VGG19	72.337	109.427	1	78.984	109.26	3	26.712	40.226	1

5.3 Cross-subject vs Cross-view

One challenge in action recognition lies in the large variations of action representations. The same action varies depending on the subject or on the view angle [29]. In this section, we compare the capacities of the neural network to treat cross subject or cross view data. The NTU RGB+D database allows two types of action classification evaluation: cross-subject evaluation and cross- view evaluation. These two kinds of evaluation allow to test the robustness of the neural network classification for data with a different view angle and data with different subjects never seen by the neural network before.

All the data representation tested in Table 1 are tried for the cross subject and cross-view simulation on all the models. Table 6 refers the data representation which reach the highest accuracy obtain with the model in cross-subject and cross-view case. Between all the models, the highest accuracy is obtained with ResNet152 in both cross-subject and cross-view configuration. The cross-subject and cross-view do not reach the highest accuracy with the same data representation. It seems more relevant for the cross-subject configuration to encode (X, Y) position and the confidence score C using 14 or 18 nodes with this order (14, 15, 16, 17), 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13. At the opposite the higher accuracy for the cross-view configuration are obtained with the encoding of (X, Y) position and the mean value of (X, Y) using 14 nodes in the 2 possible orders: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 or 0, 4, 3, 2, 1, 5, 6, 7, 10, 9, 8, 11, 12, 13.

Table 6. The highest accuracy obtained for all the models in a cross-subject and cross-view simulation and the data representation used (Table 1).

	Cross-subject		Cross-view	
Models	Accuracy	Test	Accuracy	Test
SqueezeNet1 1	75.803%	3	81.031%	4
AlexNet	74.545%	1	79.157%	5
Inception v3	81.985%	1	87.365%	5
DenseNet169	82.651%	3	88.237%	6
ResNet34	82.591%	1	88.069%	4
ResNet152	**83.317%**	1	**88.780%**	6
VGG13	79.096%	1	85.155%	6
VGG19	78.984%	3	85.885%	6

6 Comparison with the State of the Art

In this section, we compare the results obtained in the state of the art with the same database: NTU RGB+D dataset [3]. This database contains several representations of the same data sequence. We divided the methods in three categories depending on the kind of data used: skeleton (Table 7), RGB video (Table 8) or both (Table 9). Accuracy obtained using skeleton data or RGB+skeleton data is respectively 1.3% and 0.4% higher than the accuracy obtained in our study. They obtain a better accuracy, but they use more complex data containing more information including depth. Nevertheless, for action recognition based on RGB video, we obtain (at the best of our knowledge) the highest accuracy which is 83.317% with ResNet152 in cross- subject. We compare our results with other methods using the NTU RGB+D dataset even if in our case we only use 49 individual actions instead of 60 actions including 11 mutual interactions.

Table 7. Action classification performance for different architectures using skeletal data as input of the neural network

Methods using skeleton data	Date	Accuracy cross-subject	Accuracy cross-view
two streams 3DCNN [30]	August 2015	66,85%	72,58%
LSTM+CNN [31]	July 2017	82,89%	90,10%
5 CNN in parallel [6]	May 2017	82,31%	
conversion into image + CNN [7]	March 2017	79,57%	84,83%
trajectories maps+CNN [10]	December 2016	76,32%	81,08%
conversion into image + CNN [13]	July 2017	75,2%	82,1%
conversion into image +CNN [2]	March 2017	74,27%	75,74%
conversion into image + CNN [14]	April 2017	**84,6%**	**90,9%**
conversion into image +CNN [15]	April 2017	83.2%	89.3%

Table 8. Action classification performance for different architectures using RGB videos as input of the neural network

Methods using RGB videos	Date	Accuracy cross-subject	Accuracy cross-view
body part segmentation +3CNN in parallel l[32]	April 2017	80,8%	
Our method (with ResNet152)	June 2018	data representation test 1 **83,317%**	data representation test 6 **88.780%**
Our method (with DenseNet)	June 2018	data representation test 3 82,651%	data representation test 4 88.237%

Table 9. Action classification performance for different architectures using both skeletal data and RGB videos as input of the neural network

Methods using RGB videos and skeleton data	Date	Accuracy cross-subject	Accuracy cross-view
RNN+CNN [33]	March 2017	**83,74%**	**93,65%**

7 Future Research

Important issues that must still be addressed in future work are scalability of action recognition systems with respect to vocabulary size, recognition in the presence of unknown actions, scenes containing multiple persons and interactions between multiples persons. The increase of number of actions to recognize, involves the increase of the complexity of the recognition and the database size. Only actions contained in the database can be recognized. In presence of an unknown action, the classifier will return a known action class with the highest similarity. In this work, we focus on individual actions, but it should be extended to scenes containing multiples persons and interactions between multiples persons.

8 Conclusion

As artificial intelligence grows, the possibilities and applications multiply in many fields and the action recognition domain is not spared. The recent breakthroughs made in deep learning for the human pose estimation have broadened the potential of the action recognition field by getting rid of the use of dedicated motion capture sensors.

This work demonstrates that the human pose data extracted from traditional RGB videos contain sufficient information to train a classifier capable of recognizing individual actions and obtaining performances similar or superior to the state of the art. These results pave the way for motion capture databases expansion as any video can be used without the use of depth or motion sensors.

OpenPose is a very recent technological leap and to ensure that the data extracted are reasonably accurate, a comparison was carried out between OpenPose, the Kinect v2 camera and the Qualisys system. The Kinect v2 is one of the most popular motion capture systems and has been used in many researches while the Qualisys provides a reliable and precise motion capture data. We concluded that OpenPose offers sufficiently reliable 2D skeleton data to train a machine learning model.

Based on an intensive study of the related work, we chose to use an image classifier based on deep neural networks and the NTU RGB+D database as our first set of data. We converted the motion sequences into RGB images to be able to use existing deep neural network models designed for image classification. We tested several image classifier models: SqueezeNet, AlexNet, DenseNet, Resnet, Inception, VGG. Different data representations have also been tested. The data representation with the encoding of (X, Y) coordinates and the confidence score of 18 joints into RGB channels gave the highest accuracy. Finally, different image classifiers have been tested. DenseNet and ResNet are the ones which gave the highest accuracy. The highest accuracy reached during this study is 83.317% in cross-subject and 88.780% in cross-view evaluation. In the latest state-of-the-art results, higher accuracy was reached with 3D skeleton data or with the fusion of RGB data and skeleton data (+1.3% and +0.4% respectively) due to the use of more complex data containing more information including depth information.

Acknowledgment

This work was partially financed by the institutes Numediart and Infortech of the University of Mons (UMONS). This work is also partially funded by the EU and Wallonia in the framework of the ERDF/DIGISTORM project.

References

1. Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *CVPR*, 2017.
2. S. Laraba, M. Brahimi, J. Tilmanne, and T. Dutoit, "3d skeleton-based action recognition by representing motion capture sequences as 2d-rgb images," *Computer Animation and Virtual Worlds*, vol. 28, no. 3-4, 2017.
3. A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "Ntu rgb+d: A large scale dataset for 3d human activity analysis," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
4. Y. Du, Y. Fu, and L. Wang, "Skeleton based action recognition with convolutional neural network," in *Pattern Recognition (ACPR), 2015 3rd IAPR Asian Conference on*, pp. 579–583, IEEE, 2015.

5. F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Berkeley mhad: A comprehensive multimodal human action database," in *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*, pp. 53– 60, IEEE, 2013.
6. Z. Ding, P. Wang, P. O. Ogunbona, and W. Li, "Investigation of different skeleton features for cnn-based 3d action recognition," in *Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on*, pp. 617–622, IEEE, 2017.
7. Q. Ke, M. Bennamoun, S. An, F. Sohel, and F. Boussaid, "A new representation of skeleton sequences for 3d action recognition," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4570–4579, IEEE, 2017.
8. K. Yun, J. Honorio, D. Chattopadhyay, T. L. Berg, and D. Samaras, "Two-person interaction detection using body-pose features and multiple instance learning," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pp. 28–35, IEEE, 2012.
9. "Cmu dataset." http://mocap.cs.cmu.edu/. Acceded on 02-2018.
10. P. Wang, Z. Li, Y. Hou, and W. Li, "Action recognition based on joint trajectory maps using convolutional neural networks," in *Proceedings of the 2016 ACM on Multimedia Conference*, pp. 102–106, ACM, 2016.
11. "Msrc-12 kinect gesture dataset." https://www.microsoft.com/en-us/download/details.aspx?id=52283.Acceded on 02-2018.
12. V. Bloom, D. Makris, and V. Argyriou, "G3d: A gaming action dataset and real time action recognition evaluation framework," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pp. 7–12, IEEE, 2012.
13. C. Li, S. Sun, X. Min, W. Lin, B. Nie, and X. Zhang, "End-to-end learning of deep convolutional neural network for 3d human action recognition," in *Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on*, pp. 609–612, IEEE, 2017.
14. B. Li, Y. Dai, X. Cheng, H. Chen, Y. Lin, and M. He, "Skeleton based action recognition using translation- scale invariant image mapping and multi-scale deep cnn," in *Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on*, pp. 601–604, IEEE, 2017.
15. C. Li, Q. Zhong, D. Xie, and S. Pu, "Skeleton-based action recognition with convolutional neural networks," in *Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on*, pp. 597–600, IEEE, 2017.
16. [16]F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.
17. G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, vol. 1, p. 3, 2017.
18. "Skeleton return by openpose." https://arvrjourney.com/human-pose-estimation-using-openpose- with-tensorflow-part-2-e78ab9104fc8. Acceded on 02-2018.
19. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

20. K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
21. K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual networks,” in *European Conference on Computer Vision*, pp. 630–645, Springer, 2016.
22. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, *et al.*, “Going deeper with convolutions,” Cvpr, 2015.
23. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.
24. C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning.,” in *AAAI*, vol. 4, p. 12, 2017.
25. K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
26. M. Oquab, L. Bottou, I. Laptev, and J. Sivic, “Learning and transferring mid-level image representations using convolutional neural networks,” in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 1717–1724, IEEE, 2014.
27. H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, “Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning,” *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.
28. J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?,” in *Advances in neural information processing systems*, pp. 3320–3328, 2014.
29. P. Zhang, C. Lan, J. Xing, W. Zeng, J. Xue, and N. Zheng, “View adaptive neural networks for high performance skeleton-based human action recognition,” *arXiv preprint arXiv:1804.07453*, 2018.
30. H. Liu, J. Tu, and M. Liu, “Two-stream 3d convolutional neural network for skeleton-based action recognition,” *arXiv preprint arXiv:1705.08106*, 2017.
31. C. Li, P. Wang, S. Wang, Y. Hou, and W. Li, “Skeleton-based action recognition using lstm and cnn,” in *Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on*, pp. 585–590, IEEE, 2017.
32. M. Zolfaghari, G. L. Oliveira, N. Sedaghat, and T. Brox, “Chained multi-stream networks exploiting pose, motion, and appearance for action classification and detection,” in *Computer Vision (ICCV), 2017 IEEE International Conference on*, pp. 2923–2932, IEEE, 2017.
33. R. Zhao, H. Ali, and P. van der Smagt, “Two-stream rnn/cnn for action recognition in 3d videos,” *arXiv preprint arXiv:1703.09783*, 2017.