Leveraging Pre-trained CNN Models for Skeleton-Based Action Recognition

Sohaib Laraba(✉), Joëlle Tilmanne, and Thierry Dutoit

TCTS Lab, Numediart Institute, University of Mons (UMONS), Mons, Belgium
{sohaib.laraba,joelle.tilmanne,thierry.dutoit}@umons.ac.be

Abstract. Skeleton-based human action recognition has recently drawn increasing attention thanks to the availability of low-cost motion capture devices, and accessibility of large-scale 3D skeleton datasets. One of the key challenges in action recognition lies in the high dimensionality of the captured data. In recent works, researchers draw inspiration from the success of deep learning in computer vision in order to improve the performances of action recognition systems. Unfortunately, most of these studies do not leverage different available deep architectures but develop new architectures. Most of the available architecture achieve very high accuracy in different image classification problems. In this paper, we use these architectures that are already pre-trained on other image classification tasks. Skeleton sequences are first transformed into image-like data representation. The resulting images are used to train different state-of-the-art CNN architectures following different training procedures. The experimental results obtained on the popular NTU RGB+D dataset, are very promising and outperform most of the state-of-the-art results.

Keywords: Motion capture · Action recognition · Convolutional Neural Networks · Finetuning

1 Introduction

Human action recognition is an important and challenging research area in computer vision that has received much attention in the research community. It has numerous applications such as video surveillance, human-computer interaction, gaming, robotics, health, etc. Skeleton-based human action recognition has attracted increasing attention in the last decade due to wide accessibility of motion capture devices and large-scale datasets. Skeleton-based human representation considers a human body as articulated system of rigid segments connected by joints in 2D or 3D space depending on the motion capture device. Several studies have been proposed to classify 3D skeleton-based sequences [9,33,37]. Early approaches tend to build classifiers based on hand-crafted features designed manually to extract relevant information from these 3D sequences, such as relative distance between joints [10], joints orientations [29], geometric features [28], etc. These classifiers suffer from a lack of automation because of

D. Tzovaras et al. (Eds.): ICVS 2019, LNCS 11754, pp. 612–626, 2019.
https://doi.org/10.1007/978-3-030-34995-0_56

the dependency on hand-crafted features. These features are time consuming to design and extract and depend on the type of actions and data. Deep Learning (DL) has been adopted recently by the computer vision community thanks to its results that outperformed the state-of-the-art in several domains and to the availability of high-performance processing units able to train the DL models. The main advantage of deep learning, particularly in computer vision, is its ability to directly exploit raw data without any hand-crafted features. Deep learning classifiers are end-to-end systems that extract features in a fully automated way. Two deep learning methods are used in the field of action recognition: Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN). RNNs are adopted to capture temporal information from extracted spatial skeleton features. RNNs have shown their strength in language modeling [41], image captioning [38], video analysis [34] and RGB-based activity recognition [25]. CNNs in the other hand were successfully introduced for image and video classification. They learn to recognize patterns across space. A CNN will learn to recognize components of an image (lines, curves, etc.) and then learn to combine these components to recognize larger structures (objects, faces, etc.). Inspired by this success, particularly for RGB image-based action recognition, there is a growing trend of using CNNs for skeleton-based action recognition [24,39,40]. This new trend produces more accurate classifiers compared to traditional machine learning approaches. Deep learning techniques learn by creating a more abstract representation of data as the network grows deeper. As a result, the model automatically extracts features and yields higher accuracy results. Despite these good results, deep learning research in action recognition remains immature and requires more attention to produce practical systems. For example, most of the researchers explore new architectures developed specifically for this task while many new successful deep architectures are not tested in the context of action recognition. In this work, we use a spatio-temporal representation of skeleton sequences presented under the form of a form of 2D images to train CNN classifiers. We use the process of fine-tuning to leverage models already trained for classical image classification for our specific skeleton-based action recognition task. We evaluate these architectures on the public benchmark dataset NTU RGB+D [31].

2 Related Works

Several research works, addressing human action recognition from skeletal data, have been published in the last decades. In recent years, deep learning methods have been widely addressed in many fields including human action recognition. Deep learning has enabled the replacement of hand-crafted features by learned features, and the learning of whole tasks in end-to-end way. Several approaches of deep learning have been proposed for skeleton-based human action recognition and can be categorized into two main categories: Recurrent Neural Networks (RNNs) methods and Convolutional Neural Networks (CNNs) methods.

RNNs are adopted to capture temporal information from spatial sequences. Basic RNN architectures are notoriously difficult to train [4,30], and more elaborate architectures are commonly used instead, such as the LSTM (Long Short-Term Memory) [15] and the GRU (Gated Recurrent Unit) [6]. Applications of these networks have shown promising results in skeleton-based action recognition. Du et al. [8,9] designed an end-to-end hierarchical RNN architectures for skeleton-based action recognition. They divided the human skeleton into five main parts in terms of body physical structure and fed them into five independent bidirectional RNNs for local feature extraction in the first layer. In the following layers, the outputs of the RNNs were concatenated to represent the upper body and lower body, then each was further fed into another set of RNNs. The global body representation was obtained and fed to the next RNN layer. These features are fed into a fully connected layer followed by a softmax layer for classification. Veeriah et al. [36] present a differential RNN that extends LSTM structure by modeling the dynamics of states evolving over time. They proposed to add a new gating mechanism for LSTM to model the derivatives of the memory states and explore the salient action patterns. In this method, all of the input features were concatenated at each frame and were fed to the differential LSTM at each step. Shahroudy et al. [31] propose a part-aware extension of LSTM to utilize the physical structure of the human body. They split the LSTM memory cell to sub-cells to push the network towards learning the context representations for each body part separately. These methods only model the motion dynamics in the temporal domain and neglect the spatial configurations of articulated skeletons.

RNN architectures are generally used for modeling sequential data. However, one of their major drawbacks is the exploding and vanishing gradient problem and the difficulty to parallelize their training. Bai et al. [2] show that convolutional networks can perform as well as or even better that recurrent networks in many tasks such as speech recognition [42], some tasks of NLP like neural machine translation [12], classification of long sentences [1], etc.

The challenge for CNN-based methods is to effectively capture the spatio-temporal information of a skeleton sequence using image-based representation. In fact, it is inevitable to lose temporal information during the conversion of 3D information into 2D information. For example, Wang et al. [40] encode joint trajectories into texture images and utilized HSV (Hue, Saturation, Value) space to represent the temporal information. However, this method cannot distinguish some actions such as “knock” and “catch” due to the trajectory overlapping and the loss of past temporal information. Li et al. [24] propose to encode the pairwise distances of skeleton joints into texture images and encoded the pairwise distances between joints over skeleton motion sequences into color variations to capture temporal information. But this method cannot distinguish some actions of similar distance variations such as “draw-circle-clockwise” and “draw-circle-counterclockwise” due to the loss of local temporal information.

Most of these approaches transform motion capture sequences into the 2D space and develop CNN architectures for classification. However, CNNs are very

successful in image classification and a lot of pre-trained models exist and can be adapted for other tasks using fine-tuning and transfer learning techniques. These methods are very successful in many tasks such as, and in addition to image classification [13], object detection [19], person re-identification [11], biomedical image analysis [5], etc. Our proposed approach consists of transforming motion capture sequences into a simple spatio-temporal image-like representation. Then, fine-tuning of different successful state-of-the-art image classification models is applied for our task of skeleton-based action recognition.

3 Background

In this section, we cover the necessary materials for the rest of the paper. We first review Convolutional Neural Networks (CNN) and different techniques used for fine-tuning these models. Then, we propose a method to transform motion capture sequences into images. These images are fed into classical CNN models, and by the process of fine-tuning we adapt these models to our data.

3.1 Review of Convolutional Neural Networks (CNNs)

CNNs are biologically inspired models for computer vision. They have been applied for several image classification tasks such as face identification, object recognition, tumors detection and classification, etc. The architecture of a Convolutional Neural Network consists of various layers between the input and the output. Convolutional layers are the core building block of a convolutional network. They comprise of a list of n^*n filters. They are generally followed by Pooling layers that progressively reduce the spatial size of the representation and reduce the number of parameters. At the end of a CNN, there is generally a "Fully-connected" layer, similar to a multi-layer perceptron (MLP), followed by a classification layer. The classification layer can be implemented as a general-purpose classifier (for example SVM), but generally a SoftMax function is used which transforms the final vector values into probability scores between zero and one.

CNN models are trained using backpropagation algorithm. This algorithm aims to minimize a cost function that measures the total error of the model on the training dataset. Backpropagation is used to compute the gradients of the error with respect to all the weights in the network and gradient descent algorithm updates all parameters to minimize the output error. [3] gives technical details about backpropagation and gradient descent algorithms. In many situations, the amount of training data is not sufficient to adjust all parameters which causes an overfitting. In this case, transfer learning and fine-tuning are used. In these two processes, an initial step of model pre-training is used. It consists of training the CNN model on a large dataset, like ImageNet, before training on the desired dataset, and initial weights are obtained. These weights are used instead of random once and a transfer learning or fine-tuning is then applied.

– Transfer-learning (or shallow retraining) consists in freezing the pre-trained weights and only the parameters of the last classification layers need to be inferred from scratch using the new training set. This is very useful if the pre-training dataset and the final dataset are similar.
– Fine-tuning (or deep retraining) consists in freezing only few first layers or no layer, depending on the similarity of the original and final datasets, and updating all the other parameters.

In this work, six architectures (AlexNet [21], InceptionV3 [35], VGGNet [32], ResNet [14], DenseNet [16] and SqueezeNet [18]) are used with their other varieties. In total, 12 pre-trained models are used and modified to fit our classification problem (i.e. changing the last fully connected layer and the classification layer to fit the number of action classes. We compare both strategies (shallow and deep retaining) in addition to training the models from scratch (i.e. with random values).

3.2 Data Representation

In this work, we transform motion sequences, consisting of 3D joint coordinates, into RGB images to be able to use CNN models pre-trained for image classification, following the approach proposed by [22]. This approach is based on a simple transformation in which joints coordinates are normalized between 0 and 255. The normalization is done using the formula 1.

$$\begin{cases} r_i(f) = 255 * \dfrac{(x_i(f) - min(X)}{(max(X) - min(X)} \\ g_i(f) = 255 * \dfrac{(y_i(f) - min(Y)}{(max(Y) - min(Y)} \\ b_i(f) = 255 * \dfrac{(z_i(f) - min(Z)}{(max(Z) - min(Z)} \end{cases} \tag{1}$$

where:
$(x_i(f), y_i(f), z_i(f))$ are the 3D coordinates of joint i at frame f.
$min(X), min(Y), min(Z)$ are the minimum values for all joints on the 3 axes.
$max(X), max(Y), max(Z)$ are the maximum values for all joints on the 3 axes.
$(r_i(f), g_i(f), b_i(f))$ are hence the normalized values of every joint i at the frame f, where each value corresponds to the format of one channel of the RGB space.

By stacking the normalized values of all joints on all frames we obtain a long image-like representation of the motion sequence where rows correspond to joints and columns correspond to frames. The image is resized to a square representation to be easily used by different CNN architectures. Figure 1 shows some results from the NTU RGB+D dataset. The colored triangle on the bottom right of the figure can be considered as a dictionary to interpret these images. The red color represents the X coordinate, the green color represents the Y coordinate and the blue color represents the Z coordinate. For example, the greener the part

of an image from left to right means that the Y value is getting higher. We can see for example at the images corresponding to the action "Throw" the lines on the top and in the middle (corresponding to right and left arms joints) are getting in particular more of the green color at a certain part of the image then it comes back to the original color. This is the translation of the arms moving up to throw the object before going down again.

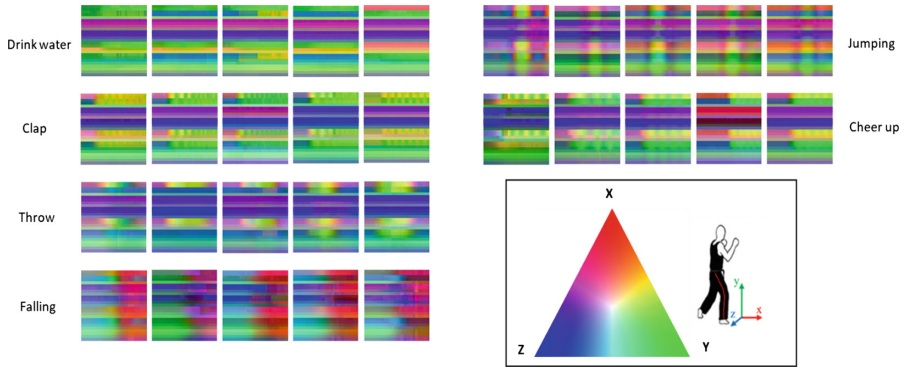

Fig. 1. Examples of motion capture sequences represented into RGB images. (Color figure online)

In the images corresponding to "Falling", we can see, starting from the middle, that the images are getting redder to purple. This can be explained by the fact that, when the person falls, all the joints Y values are getting smaller (hence less green). In order to be invariant to global position, all 3D joint coordinates are normalized with respect to the "SpineBase" joint (the most stable joint from

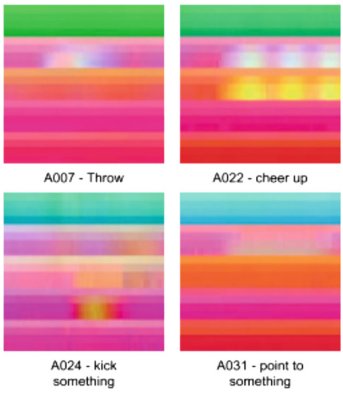

Fig. 2. Examples of the motion capture sequences transformed into images after pre-processing.

the Kinect 2). 3D joint coordinates are hence relative coordinates. The order of the joints is also modified in order to have more present visual patterns, and 4 very noisy joints have been discarded (thumbs and hand tips joints). The final order of joints is the following: 3, 2, 20, 1, 0, 8, 9, 10, 11, 4, 5, 6, 7, 16, 17, 18, 19, 12, 13, 14, 15 (the joints' numbers are shown in Fig. 4.). Figure 2 shows some examples of the final representation. These images will be fed to multiple CNN models to select the best one in terms of accuracy.

4 Experimental Results

In this section, the proposed method is evaluated on the NTU RGB+D dataset. We evaluate different state-of-the-art CNN architectures using different training strategies An overview of the end-to-end CNN system used in this work is illustrated in Fig. 3. After the skeletons are pre-processed, motion capture sequences are transformed into image data representations. These images are fed into different CNN architectures to be trained using different strategies. The CNN part of our models generate automatically features (feature maps) that are fed into a fully connected network for classification.

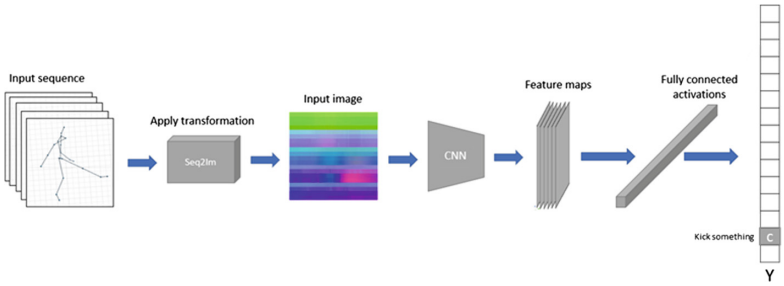

Fig. 3. Illustration of the proposed End-to-end system for skeleton-based action recognition.

4.1 Data Structure

In order to compare our results with existing works, we evaluate our method on the NTU RGB+D dataset. This dataset was collected using three Kinect V2 sensors at the same time covering three views (45°, 0°, 45°) and contains more than 56,000 action sequences. A total of 60 different action classes are performed by 40 subjects aged from 10 to 35 years. Among these action classes, 49 are performed by single persons and 11 are interactions between two people. Only the 49 single person actions were used in our tests. In addition to depth maps, RGB frames, and infrared (IR) sequences, information of 25 3D joints are available. This dataset is challenging because of the large intraclass and

viewpoint variations; however, due to its large scale, it is highly suitable for deep learning. The captured Kinect V2 skeletons have 25 joints in total. The configuration and the given order of joints is shown in Fig. 4.

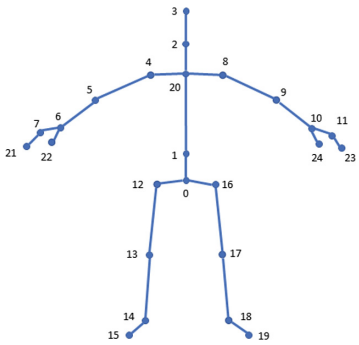

Fig. 4. Configuration of the skeleton given by the Kinect V2.

4.2 Training the Deep Network

In this experiment, 12 state-of-the-art architectures (AlexNet, Inception V3, VGG11, VGG16, VGG19, ResNet34, ResNet50, ResNet152, DenseNet121, DenseNet169, DenseNet201 and SqueezeNet) are trained on the dataset described in the previous section. The numbers after the architectures' names represent the number of layers. This will allow us also to analyze the effect of the depth of the models as well. We trained these architectures using three different strategies. The first strategy consists in training the CNN from scratch starting from random weights. The other two strategies are based on transfer learning using pre-trained networks. The first transfer learning approach, called shallow retraining approach, consists in fine-tuning only the last added fully connected layer, while the rest of the network is used as feature extractor. The second approach, called deep retraining approach, fine-tunes all the network layers. All the 36 training configurations use the same hyperparameters (momentum 0.9, weight decay 0.0005, learning rate 0.001, batch size 30 and a number of epochs of 15). The dataset is divided following two evaluation protocols:

- Cross-subject evaluation: the 40 subjects are split into training and testing groups. Each group consists of 20 subjects.
- Cross-view evaluation: the samples of one camera (corresponding to one viewpoint) are used for testing and samples of the two other cameras are used for training.

All experiments are performed on a machine having the specifications summarized in Table 1.

Table 1. Training and testing machine specifications.

	Characteristics
Memory (RAM)	32 GB
Processor (CPU)	Intel® Core™ i7-7800X CPU @ 3.50 GHz 12
Graphics (GPU)	2 * GeForce GTX 1080 Ti/PCIe/SSE2 (11 GB)
Operating system	Linux Ubuntu 16.04 64 bits

4.3 Models Evaluation: Results and Discussion

In this section, we discuss the results obtained for different experiments. The accuracy and the training time for all the 12 models are shown in Tables 2 and 3 for cross-subject and cross-view evaluation protocols respectively.

Cross-Subject Evaluation Protocol. From Table 2, we get the highest scores of accuracy with the models ResNet50 and DenseNet201, around 82%. This accuracy is obtained using the deep retraining approach. Analyzing this table, we notice that the lowest scores are obtained using the shallow retraining approach. Such low scores are linked to the fact that we retrain only the last added layer and we keep the rest of the model untouched. The convolutional layers in this situation are used as feature extractors. This can work and give good results in cases where we have images that are similar to the original dataset that was used to pre-train the model (ImageNet). However, our images are more abstract and completely different from the original dataset. The features extracted are hence meaningless in regard to our data.

Table 2. Comparison of different proposed models for cross-subject evaluation.

Model	From scratch		Retrain shallow		Retrain deep	
	Time (h)	Acc	Time (h)	Acc	Time (h)	Acc
AlexNet	1.17	0.6544	1.09	0.2371	1.76	0.7319
InceptionV3	1.78	0.6607	0.61	0.2524	1.68	0.7953
VGG11	2.33	0.6841	1.95	0.2627	3.29	0.7691
VGG16	2.82	0.6674	2.15	0.2696	3.87	0.7760
VGG19	3.07	0.6490	2.27	0.2200	3.70	0.7514
ResNet34	1.05	0.7202	0.85	0.3768	1.05	0.8020
ResNet50	1.31	0.6809	0.97	0.3835	1.30	0.8207
ResNet152	2.45	0.6756	2.09	0.3910	2.43	0.8118
DensNet121	1.29	0.7468	1.53	0.4228	1.28	0.8096
DensNet169	1.56	0.7610	1.49	0.4380	1.53	0.8172
DeseNet201	1.86	0.7622	1.84	0.4422	1.82	0.8200
SqueezeNet	0.61	0.6573	0.55	0.3145	0.62	0.7209

We can notice relatively higher scores, compared to the shallow retraining, in the case of the retraining from scratch (from random weights). Retraining the whole model allowed it to develop features that are adapted to our data. Nevertheless, the scores are not as high as the deep retraining approached because retraining from scratch may need more data and more time to converge. Transfer learning using deep retraining allowed a quicker convergence. We can also conclude that more layers do not automatically mean higher accuracy. It can be the opposite in many cases like VGG and ResNet. VGG architecture with 19 layers for example gives lower accuracy than the one with 11 and 16 layers in a deep retraining approach. This can be explained by the fact that the more layers we have, the more risk we have of overfitting. We can finally notice that SqueezeNet gives relatively high scores, particularly in a deep retraining approach. SqueezeNet is a very small network with few parameters. It has a total size of less than 0.5 MB. Compared to AlexNet for example that has a total size of 240 Mb, it makes it easy and practical to fit into embedded systems and smartphones. Evaluation of the models' performances comparing training time and accuracy is displayed in Fig. 5.

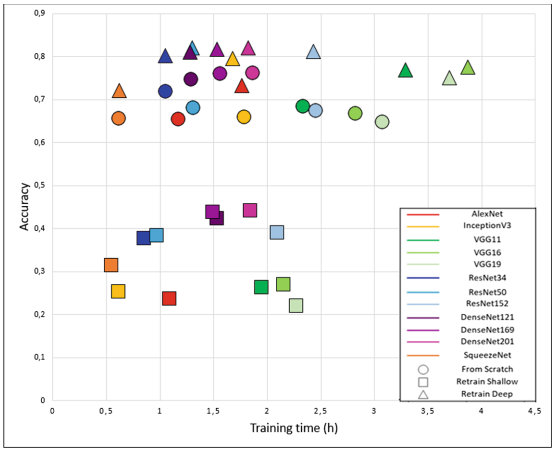

Fig. 5. Cross-subject evaluation: accuracy vs. training time (h).

Cross-View Evaluation Protocol. From Table 3, we get the highest scores in the "Cross-View" evaluation protocol with the models ResNet34, ResNet152 and DenseNet201 of about 86%. The exact same conclusions as in "Cross-Subject" evaluation can be drawn.

Table 3. Comparison of different proposed models for cross-view evaluation.

Model	From scratch		Retrain shallow		Retrain deep	
	Time (h)	Acc	Time (h)	Acc	Time (h)	Acc
AlexNet	0.92	0.6400	1.18	0.2267	1.15	0.7486
InceptionV3	1.73	0.6482	0.61	0.2573	1.54	0.8046
VGG11	1.75	0.7573	2.13	0.2693	2.27	0.8144
VGG16	2.23	0.7635	2.18	0.2530	2.75	0.8269
VGG19	2.50	0.7643	2.25	0.2160	2.99	0.8001
ResNet34	1.01	0.6997	0.83	0.3672	1.03	0.8600
ResNet50	1.27	0.6457	0.96	0.3774	1.29	0.8592
ResNet152	2.67	0.7375	1.61	0.3906	2.38	0.8611
DensNet121	1.30	0.7877	0.94	0.4104	1.27	0.8550
DensNet169	1.64	0.7767	1.10	0.4302	1.51	0.8489
DeseNet201	1.95	0.7924	1.27	0.4422	1.80	0.8654
SqueezeNet	0.4	0.7021	0.57	0.3065	0.62	0.7760

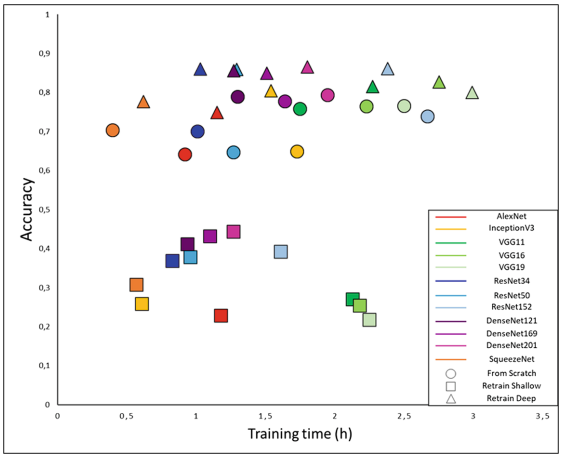

Fig. 6. Cross-view evaluation: accuracy vs. training time (h).

Evaluation of the models' performances by comparing training time and accuracy is also displayed in Fig. 6, which is similar to the previous evaluation scenario (cross-subject). The obtained results can be compared to several works in the state of the art that used the same dataset (Table 4). The highest score we obtained is 82,07% using ResNet50 in the cross-subject evaluation protocol and 86,54% using DenseNet201 in the cross-view evaluation protocol. Our results outperform most of state-of-the-art methods in both cross-subject and cross-view protocols. Our results are obtained by transforming motion sequences

Table 4. Comparison of different proposed models for cross-view evaluation.

Method	Cross-subject	Cross-view
Using CNNs		
two streams 3DCNN [26]	66,85%	72,58%
conversion into image + CNN [20]	79,57%	84,83%
trajectories maps + CNN [40]	76,32%	81,08%
conversion into image + CNN [23]	75,20%	82,10%
conversion into image + CNN [22]	74,27%	75,74%
conversion into image + CNN [7]	**83.2%**	**89.3%**
Using other DL methods		
HBRNN [9]	59,07%	63,97%
Deep RNN [31]	56.29%	64.09%
Deep LSTM [31]	60.69%	67.29%
PA-LSTM [31]	62.93%	70.27%
LieNet [17]	61.37%	66.95%
ST-LSTM [27]	69.20%	77.70%
Our method	**82.07%**	**86.54%**

into images and using pre-trained models without developing new architectures. Specifically, we improve accuracy by 3 to 8% compared to other techniques using CNNs with image-like transformation of motion sequences. This proves that adapting the weights from pre-trained models for a new task improves the performance of deep learning networks and gives high scores even if the original task is very different.

The work done by Li et al. [7] have significantly higher results. Authors have developed a new architecture dedicated for this task, using two-stream CNNs and achieved 83.2% of accuracy for cross-subject evaluation and 89.3% for cross-view evaluation.

5 Conclusion

In this work, we studied the use of pre-trained convolutional neural networks for skeleton-based action recognition. In order to handle the high dimensionality of skeleton sequences, we transformed them into RGB images so they can be fed directly to different CNN models that are designed for image classification. We exploited different state-of-the-art pre-trained architectures and used a process called "fine-tuning" to adapt the weights to our task. Even though the obtained images are abstract and completely different from original dataset (ImageNet) used to pre-train different models, we achieved high classification scores and outperformed most of action classification state-of-the-art results on the NTU RGB+D dataset. This proves that, firstly, the image-like representation

of motion capture skeleton sequences can be well interpreted by different CNN architectures. This also proves that using already trained classification models helps in other classification tasks and accelerates the model's convergence. Nonetheless, that are still quite a few open issues. For example, studying the use of rotation in addition to 3D joint locations and exploiting other pre-trained architectures that use both CNNs and RNNs for our task. Besides, building a CNN architecture for our specific task, pre-training it on different motion capture datasets, then apply fine-tuning for the desired dataset, could be more effective. Last but not least, studying the behavior of different CNN models using visualization techniques to set the most optimized parameters would also be an interesting direction.

References

1. Adel, H., Schütze, H.: Exploring different dimensions of attention for uncertainty detection. arXiv preprint arXiv:1612.06549 (2016)
2. Bai, S., Kolter, J.Z., Koltun, V.: An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271 (2018)
3. Bengio, Y., Goodfellow, I., Courville, A.: Deep learning, vol. 1. Citeseer (2017)
4. Bengio, Y., Simard, P., Frasconi, P., et al.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. **5**(2), 157–166 (1994)
5. Broadwater, D.R., Smith, N.E.: A fine-tuned inception v3 constitutional neural network (CNN) architecture accurately distinguishes between benign and malignant breast histology. Technical report, 59 MDW San Antonio United States (2018)
6. Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: encoder-decoder approaches. arXiv preprint arXiv:1409.1259 (2014)
7. Du, Y., Fu, Y., Wang, L.: Skeleton based action recognition with convolutional neural network. In: 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), pp. 579–583. IEEE (2015)
8. Du, Y., Fu, Y., Wang, L.: Representation learning of temporal dynamics for skeleton-based action recognition. IEEE Trans. Image Process. **25**(7), 3010–3022 (2016)
9. Du, Y., Wang, W., Wang, L.: Hierarchical recurrent neural network for skeleton based action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1110–1118 (2015)
10. Efros, A.A., Berg, A.C., Mori, G., Malik, J.: Recognizing action at a distance. In: Null, p. 726. IEEE (2003)
11. Fan, H., Zheng, L., Yan, C., Yang, Y.: Unsupervised person re-identification: clustering and fine-tuning. ACM Trans. Multimedia Comput. Commun. Appl. (TOMM) **14**(4), 83 (2018)
12. Gehring, J., Auli, M., Grangier, D., Dauphin, Y.N.: A convolutional encoder model for neural machine translation. arXiv preprint arXiv:1611.02344 (2016)
13. Han, D., Liu, Q., Fan, W.: A new image classification method using CNN transfer learning and web data augmentation. Expert Syst. Appl. **95**, 43–56 (2018)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

15. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
16. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
17. Huang, Z., Wan, C., Probst, T., Van Gool, L.: Deep learning on lie groups for skeleton-based action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6099–6108 (2017)
18. Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J., Keutzer, K.: SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size. arXiv preprint arXiv:1602.07360 (2016)
19. Kang, K., et al.: T-CNN: tubelets with convolutional neural networks for object detection from videos. IEEE Trans. Circ. Syst. Video Technol. **28**(10), 2896–2907 (2017)
20. Ke, Q., Bennamoun, M., An, S., Sohel, F., Boussaid, F.: A new representation of skeleton sequences for 3D action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3288–3297 (2017)
21. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
22. Laraba, S., Brahimi, M., Tilmanne, J., Dutoit, T.: 3D skeleton-based action recognition by representing motion capture sequences as 2D-RGB images. Comput. Anim. Virtual Worlds **28**(3–4), e1782 (2017)
23. Li, C., Sun, S., Min, X., Lin, W., Nie, B., Zhang, X.: End-to-end learning of deep convolutional neural network for 3D human action recognition. In: 2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pp. 609–612. IEEE (2017)
24. Li, C., Hou, Y., Wang, P., Li, W.: Joint distance maps based action recognition with convolutional neural networks. IEEE Signal Process. Lett. **24**(5), 624–628 (2017)
25. Li, Q., Qiu, Z., Yao, T., Mei, T., Rui, Y., Luo, J.: Action recognition by learning deep multi-granular spatio-temporal video representation. In: Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval, pp. 159–166. ACM (2016)
26. Liu, H., Tu, J., Liu, M.: Two-stream 3D convolutional neural network for skeleton-based action recognition. arXiv preprint arXiv:1705.08106 (2017)
27. Liu, J., Shahroudy, A., Xu, D., Wang, G.: Spatio-temporal LSTM with trust gates for 3D human action recognition. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 816–833. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_50
28. Müller, M.: Information Retrieval for Music and Motion, vol. 2. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74048-3
29. Ohn-Bar, E., Trivedi, M.: Joint angles similarities and HOG2 for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 465–470 (2013)
30. Pascanu, R., Mikolov, T., Bengio, Y.: On the difficulty of training recurrent neural networks. In: International Conference on Machine Learning, pp. 1310–1318 (2013)
31. Shahroudy, A., Liu, J., Ng, T.T., Wang, G.: NTU RGB+ D: a large scale dataset for 3D human activity analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1010–1019 (2016)

32. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
33. Song, S., Lan, C., Xing, J., Zeng, W., Liu, J.: An end-to-end spatio-temporal attention model for human action recognition from skeleton data. In: Thirty-first AAAI Conference on Artificial Intelligence (2017)
34. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using LSTMs. In: International Conference on Machine Learning, pp. 843–852 (2015)
35. 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)
36. Veeriah, V., Zhuang, N., Qi, G.J.: Differential recurrent neural networks for action recognition. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4041–4049 (2015)
37. Vemulapalli, R., Arrate, F., Chellappa, R.: Human action recognition by representing 3D skeletons as points in a lie group. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 588–595 (2014)
38. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: a neural image caption generator. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156–3164 (2015)
39. Wang, P., Li, W., Li, C., Hou, Y.: Action recognition based on joint trajectory maps with convolutional neural networks. Knowl.-Based Syst. **158**, 43–53 (2018)
40. Wang, P., Li, Z., Hou, Y., Li, W.: Action recognition based on joint trajectory maps using convolutional neural networks. In: Proceedings of the 24th ACM International Conference on Multimedia, pp. 102–106. ACM (2016)
41. Xu, K., et al.: Show, attend and tell: neural image caption generation with visual attention. In: International Conference on Machine Learning, pp. 2048–2057 (2015)
42. Zhang, Y., Pezeshki, M., Brakel, P., Zhang, S., Bengio, C.L.Y., Courville, A.: Towards end-to-end speech recognition with deep convolutional neural networks. arXiv preprint arXiv:1701.02720 (2017)