Learning to Recognize 3D Human Action from A New Skeleton-based Representation Using Deep Convolutional Neural Networks

Huy-Hieu Pham1,2,∗, Louahdi Khoudour1, Alain Crouzi2, Pablo Zegers3, Sergio A. Velastin4,5,6
1 Cerema, Equipe-projet STI, 1 Avenue du Colonel Roche, 31400, Toulouse, France
2 Institut de Recherche en Informatique de Toulouse (IRIT), Université de Toulouse, UPS, 31062 Toulouse, France
3 Aparnix, La Gioconda 4355, 10B, Las Condes, Santiago, Chile
4 Department of Computer Science, Applied Artificial Intelligence Research Group, University Carlos III de Madrid, 28270 Madrid, Spain
5 School of Electronic Engineering and Computer Science, Queen Mary University of London, UK
6 Cortexica Vision Systems Ltd., London, UK
∗E-mail: huy-hieu.pham@cerema.fr

Abstract: Recognizing human actions in untrimmed videos is an important challenging task. An effective 3D motion representation and a powerful learning model are two key factors influencing recognition performance. In this paper we introduce a new skeleton-based representation for 3D action recognition in videos. The key idea of the proposed representation is to transform 3D joint coordinates of the human body carried in skeleton sequences into RGB images via a color encoding process. By normalizing the 3D joint coordinates and dividing each skeleton frame into five parts, where the joints are concatenated according to the order of their physical connections, the color-coded representation is able to represent spatio-temporal evolutions of complex 3D motions, independently of the length of each sequence. We then design and train different Deep Convolutional Neural Networks (D-CNNs) based on the Residual Network architecture (ResNet) on the obtained image-based representations to learn 3D motion features and classify them into classes. Our method is evaluated on two widely used action recognition benchmarks: MSR Action3D and NTU-RGB+D, a very large-scale dataset for 3D human action recognition. The experimental results demonstrate that the proposed method outperforms previous state-of-the-art approaches whilst requiring less computation for training and prediction.

1 Introduction

Human Action Recognition (HAR) [1] is an important topic in machine vision. HAR methods have been widely applied in building electronic systems based on machine intelligence such as intelligent surveillance [2, 3], human-machine interaction [4, 5], health care [6], and so on. Although significant progress has been made in the last few years, HAR is still a challenging task due to many obstacles, e.g. camera viewpoint, occlusions, or large intra-class variations [7].

There are two main types of intelligence that are artificial intelligence and machine intelligence. HAR can be considered as an electronic system based on machine intelligence. The first stage of any HAR system is data acquisition. Nowadays, a variety of electronic imaging systems can be used for this task such as traditional optical cameras, RGB-D sensors, Thermal Infrared (IR) sensors, and Synthetic Aperture Radar (SAR) sensors. For instance, IR sensors have been exploited in HAR [8]. These sensors are able to generate images based on the heat radiated by the human body and to work independently of lighting conditions. SAR imaging has also been widely used for detecting and analyzing human activities [9]. This type of sensor is able to operate far away from potential targets and functions during the daytime as well as nighttime, under all weather conditions. In particular, SAR Image Segmentation (SIS) techniques [10–12] can segment humans from other components (e.g. objects, background) in the scene. As a result, the use of SAR sensors offers big advantages for human action recognition, in particular for human target detection and identification in a complex environment such as in military applications.

Traditional studies on HAR mainly focus on the use of RGB sequences provided by 2D cameras. These approaches typically recognize actions based on hand-crafted local features such as Cuboids [13], HOG/HOF [14], HOG-3D [15], which are extracted from the appearance and movements of human body parts. However, it is very hard to fully capture and model the spatial-temporal information of human action due to the absence of 3D structure from the scene. Therefore, RGB-D cameras are becoming one of the most commonly-used sensors for HAR [16–18]. Recently, the rapid development of cost-effective and easy-to-use depth cameras, e.g. Microsoft Kinect™ sensor [19, 20], ASUS Xtion PRO [21], or Intel® Realsense™ [22] has helped dealing with problems related to 3D action recognition. In general, depth sensors are able to provide detailed information about the 3D structure of human body. Therefore, many approaches have been proposed for 3D HAR based on data provided by depth sensors such as RGB sequences, depth, or combining these two data types (RGB-D). Furthermore, these devices have integrated real-time skeleton tracking algorithms [23] that provide high-level information about human motions in a 3D space. Thus, exploiting skeletal data for 3D HAR is an active research topic in computer vision [24–28].

In recent years, approaches based on Convolutional Neural Networks (CNNs) have achieved outstanding results in many image recognition tasks [29–31]. After the success of AlexNet [29], a new direction of research has been opened for designing and optimizing higher performing CNN architectures. As a result, there are substantial evidences that seem to show that the learning performance of CNNs may be significantly improved by increasing the number of hidden layers [32, 33]. In the literature of HAR, many studies have indicated that CNNs have the ability to learn complex motion features better than hand-crafted approaches. However, most previous works have just focused on exploring simple CNNs such as AlexNet [29] and have not exploited the potential of recent state-of-the-art very deep CNN architectures (D-CNNs), e.g. Residual Networks (ResNet) [34]. In addition, most existing CNN-based approaches limit themselves to using RGB-D sequences as the input to learning models. Although RGB-D images are informative for understanding human action, the computation complexity of learning models...
increases rapidly when the dimension of the input features is large. Therefore, representation learning models based on RGB-D modality become more complex, slower and less practical for solving large-scale problems as well as real-time applications.

Different from previous works, to take full advantages of 3D skeletal data and the learning capacity of D-CNNs, this paper proposes an end-to-end deep learning framework for 3D HAR from skeleton sequences. We focus on solving two main issues: First, using a simple skeleton-to-image encoding method to transform the 3D coordinates of the skeletal joints into RGB images. The encoding method needs to ensure that the image-based representation of skeleton sequences is able to effectively represent the spatial structure and temporal dynamics of the human action. Second, we design and train different D-CNNs based on Residual Networks (ResNets) – a recent state-of-the-art CNN for image recognition, to learn and classify actions from the obtained image-coded representations.

This paper is an extended version of our work published in the 8th International Conference of Pattern Recognition Systems (ICPRS 2017) [35]. It is part of our research project about investigating and developing a low-cost system based on RGB-D data provided by depth sensors for understanding human actions and analyzing their behaviors in indoor environments such as inside home, offices, residential buildings, buses, trains, etc. In general, three observations motivate our exploration of using ResNet [34] for 3D HAR from skeleton sequences, including: (1) human actions can be correctly represented as the movements of skeletons [27] and the spatial-temporal dynamics of skeletons can be transformed into 2D image structure, which can be effectively learned by D-CNNs; (2) skeletal data is high-level information with much less complexity than RGB-D sequences, this advantage makes our action learning model much simpler and faster; (3) many evidences [32, 33] show that deeper convolutional model, especially ResNet [34] can boost the learning accuracy in image recognition tasks.

We evaluate the proposed method on two benchmark skeleton datasets (i.e. MSR Action3D [36] and NTU-RGB+D [37]). Experimental results confirmed the above statements since our method achieved state-of-the-art performance compared with the existing results using the same evaluation protocols. Furthermore, we also indicate the effectiveness of our learning framework in terms of computational complexity.

In summary, two main contributions of this paper include:

- First, we introduce a new skeleton-based representation (Sec. 3.1) and an end-to-end learning framework *(Sec. 3.2)* based on D-CNNs to learn the spatio-temporal evolutions of 3D motions and then to recognize human actions;
- Second, we show the effectiveness of our method on HAR tasks by achieving state-of-the-art performance on two benchmark datasets, including the most challenging skeleton benchmark currently available for 3D HAR [37] (Sec. 5).

The advantage of the proposed method is that it has high computational efficiency (Sec. 5.4). Moreover, this approach is general and can be easily applied to other time-series problem, e.g. recognizing human actions with mobile devices with integrated inertial sensors.

The rest of the paper is organized as follows: Section 2 discusses related works. In Section 3, we present the details of our proposed method, including the color-encoding process from skeletons to RGB images and deep learning networks. Datasets and experiments are described in Section 4. Experimental results are reported in Section 5 with a detailed analysis of computational efficiency. Finally, Section 6 concludes the paper with a discussion on the future work.

*Codes and pre-trained models will be shared to the community at https://github.com/huyhieupham/ after publication.*

2 Related Work

Skeleton-based Action Recognition (SBAR) using depth sensors has been widely studied in recent years. In this section, we briefly review existing SBAR methods, including two main categories that are directly related to our work. The first category is approaches based on hand-crafted local features of skeletons. The second is approaches based on deep learning networks, especially Recurrent Neural Networks (RNN-LSTMs).

**Approaches based on hand-crafted local features:** Human motion can be considered as a spatio-temporal pattern. Many researchers have built hand-crafted feature representations for SBAR and then used temporal models for modeling 3D human motion. For instance, Wang et al. [24] represented the human motion from skeletal data by means of the pairwise relative positions of the key joints. The authors then employed Hidden Markov Model (HMM) [38] for modeling temporal dynamics of actions. A similar approach has been presented by Lv et al. [39] where the 3D joint position trajectories have been mapped into feature spaces. The dynamics of actions was then learned by one continuous HMM. Xia et al. [25] proposed the use of histogram-based representation of human pose, and then actions were classified into classes by a discrete HMM. Wu and Shao [40] also exploited HMM for recognizing actions from high-level features of skeletons. Venulapalli et al. [27] represented the 3D geometric interactions of the body parts in a Lie Group. These elements were processed with a Fourier Temporal Pyramid (FTP) transformation for capturing temporal evolutions of the original motions. Different from the studies above, Luo et al. [41] proposed a new dictionary learning model to learn the spatio-temporal information from skeleton sequences. Although promising results have been achieved from approaches based on hand-crafted local features and probabilistic graphical models, they have some limitations that are very difficult to overcome. E.g., most of these approaches are data-dependent and require a lot of hand-designing features. HMM-based methods require preprocessing input data in which the skeleton sequences need to be segmented or aligned. Meanwhile, FTP-based approaches can only utilize limited contextual information of actions.

**Approaches based on deep learning:** Recurrent Neural Networks with Long Short-Term Memory Network (RNN-LSTMs) [42, 43] are able to model the contextual information of the temporal sequences as skeleton data. Thus, many authors have explored RNN-LSTMs for SAR. For instance, Du et al. [44] proposed an end-to-end hierarchical RNN-LSTM for modeling local motions of a body part in which all skeleton frames were divided into five parts according to the human physical structure. Each part of a skeleton was then fed into an independent RNN and then fused to be the inputs of higher layers. The final representation was used for the classification task. Zhu et al. [45] proposed a LSTM network with a mixed-norm regularization term to cost functions in order to learn the co-occurrence of discriminative features from skeletal data. Liu et al. [46] introduced a new gating mechanism with LSTM network to analyze the reliability of input skeleton sequences. In another study, Shahroudy et al. [37] used a part-aware LSTM network in which the memory-cell was divided into sub-cells such that each sub-cell can model long-term contextual representation of a body part. Finally, all these sub-cells were concatenated to the final output. Although RNN-LSTMs are able to model the long-term temporal of motion and experimental results provided that RNN-LSTMs have outperformed many other approaches, RNN-based approaches just consider skeleton sequences as a kind of low-level feature by feeding directly the raw 3D joint coordinates carried in skeletons into the network input. The huge number of input parameters may make RNNs become very complex and easily lead to overfitting due to insufficient training data. Moreover, many RNN-LSTMs act as a classifier and cannot extract high-level features [47] for recognition tasks.

Several authors have exploited the feature learning ability of CNNs on skeletal data [48, 49]. However, such studies mainly focus on the use of complex encoding methods for finding good skeletal
representations and learning geometric features carried in skeleton sequences with simple CNN architectures. In contrast, in this paper we concentrate on proposing a new and simple skeleton-based representation and exploiting the power of D-CNNs for action recognition. Our experiments on two public datasets, including the MSR Action3D dataset [36] and the NTU-RGB+D dataset [37] show state-of-the-art performance on both datasets. Moreover, in terms of computational cost, our measurement and analyses show that the proposed model is fast enough for many real-time applications.

3 Proposed Method

This section presents the proposed method. We first describe the color-encoding process that allows us to represent the spatial structure and temporal dynamics of skeleton sequences as the static structure of a 2D color image. We then review the main idea of ResNet [34] and propose five different ResNet configurations, including 20-layer, 32-layer, 44-layer, 56-layer, and 110-layer networks to learn motion features carried in the image-based representations and perform action classification.

3.1 Building image-based representation from skeleton sequences

Currently, skeletal data that contains 3D joint coordinates can be obtained from depth cameras via real-time skeleton estimation algorithms [50, 51]. This technology allows to extract the position of the key joints in the body, which is suitable for 3D HAR problems. E.g., the latest version of the Microsoft Kinect TM sensor [19, 20] (Kinect v2 sensor) can track the main 25 joints of human body in real-time speed as shown in Figure 1.

Fig. 1: The position of 25 joints in the human body extracted by Kinect v2 sensor [37]. Skeleton sequences can be recorded frame-by-frame at 30 FPS.

In this paper, instead of feeding raw 3D joint coordinates directly to RNN-LSTMs as in many previous works [37, 44–46], we propose the use of D-CNNs for learning motion features from skeleton sequences. The first step is to find a skeleton-based representation that can effectively capture the spatio-temporal evolutions of skeletons and that can also be easily learned by learning method as D-CNNs. One solution for this problem is to transform each skeleton sequence into a single color image (e.g., a RGB image) in which the pixel values must have the ability to represent the movement of skeletons.

To this end, we transform the 3D joint coordinates of each skeleton into a new space by normalizing their coordinates using a transformation function. Specifically, given a skeleton sequence \( S \) with \( N \) frames \( \{F_1, F_2, ..., F_N\} \), let \((x_i, y_i, z_i)\) be the 3D coordinates of each joint in frame \( \{F_n\} \in S, n \in [1, N] \). A normalization function \( F(\cdot) \) is used to transform all 3D joint coordinates to the range of \([0, 255]\) as follows:

\[
(x'_i, y'_i, z'_i) = F(x_i, y_i, z_i)
\]

(1)

\[
x'_i = 255 \times \frac{x_i - \min\{\tau\}}{\max\{\tau\} - \min\{\tau\}}
\]

(2)

\[
y'_i = 255 \times \frac{y_i - \min\{\tau\}}{\max\{\tau\} - \min\{\tau\}}
\]

(3)

\[
z'_i = 255 \times \frac{z_i - \min\{\tau\}}{\max\{\tau\} - \min\{\tau\}}
\]

(4)

where \((x'_i, y'_i, z'_i)\) is 3D joint coordinate in the normalized space \( \tau \). The \( \max\{\tau\} \) and \( \min\{\tau\} \) are the maximum and minimum values of all coordinates, respectively. To preserve the spatio-temporal information of skeleton movements, we stack all normalized frames according to the temporal order \( S' = \{F_1, F_2, ..., F_N\} \) to represent the whole action sequence. These elements are quantified to RGB color space and can be stored as RGB images. In this way, we convert the skeletal data to 3D tensors that will then be fed into deep learning networks as the input for feature leaning and classification. Figure 2 illustrates the skeleton-to-image transformation process.

Fig. 2: Illustration of the data transformation process. Each skeleton sequence is encoded into a single color image that represents the spatio-temporal evolutions of the motion. Here, \( N \) denotes the number of frames in each sequence while \( K \) denotes the number of joints in each frame. The value of \( K \) depends on each RGB+D dataset. E.g., \( K \) is equal to 20 for MSR Action3D dataset [36]. Meanwhile, \( K \) is equal to 25 for NTU-RGB+D dataset [37].

Naturally, the human body is structured as four limbs and one trunk (see Figure 1). Simple actions can be performed through the movement of a limb, e.g., hand-waving, kicking forward, etc. More complex actions combine the movements of a group of limbs or the whole body, e.g., running or swimming. To keep the local characteristics of the human action [53, 54], we divide each skeleton into five parts, including two arms \( (P_1, P_2) \), two legs \( (P_3, P_4) \), and one trunk \( (P_5) \). In each part, the joints are concatenated according their physical connections as shown in Figure 3. These parts are then

Fig. 3: Map of joints in each part from \( P_1 \) to \( P_5 \) on a skeleton provided by Microsoft Kinect v2 TM sensor [52].
connected in a sequential order: \( P_1 \rightarrow P_2 \rightarrow P_4 \rightarrow P_1 \rightarrow P_5 \). The whole process of rearranging all frames in a sequence can be done by rearranging the order of rows of pixels in the color-based representation. This process is illustrated in Figure 4.

\[ \text{Fig. 4: Rearranging the order of joints according to the human body physical structure.} \]

Like that, we have encoded skeleton sequences into RGB images. Figure 5 shows some examples of RGB images obtained from input sequences of MSR Action3D dataset [36]. These images will be learned and classified by D-CNN models. This way, the original skeleton sequence will be recognized via the corresponding image.

\[ \text{Fig. 5: The image-based representations obtained from several samples of MSR Action3D dataset [36]. In our experiments, all images are resized to } 32 \times 32 \text{ pixels. Best viewed in color.} \]

### 3.2 Deep residual networks for skeleton-based human action recognition

In this section, we introduce our network designs for recognizing human action from image-based representation obtained in Section 3.1. Before that, in order to put our method into context, we will briefly review the key idea of ResNet [34].

#### 3.2.1 Residual learning method

Very deep neural networks, especially D-CNNs have demonstrated to have a high performance on many visual-related tasks [32, 55]. However, D-CNNs are very difficult to optimize due to the vanishing gradients problem. If the network is deep enough, the error signal from the output layer can be completely attenuated on its way back to the input layer. In addition, another problem called the “degradation phenomenon” [56] also impedes the convergence of deeper networks. More specially, adding more layers to a deep network can lead to higher training/testing error. To overcome these challenges, ResNet [34] has been introduced. The key idea behind ResNet architectures is the presence of shortcut connections between input and output of each ResNet building block. These shortcut connections are implemented by identity mappings and are able to provide a path for gradients to back propagate to early layers in the network. This idea improves the information flow in ResNets and helps them to learn faster.

Mathematically, a layer or a series of layers in a typical CNN model will try to learn a mapping function \( y = \mathcal{F}(x) \) from input feature \( x \). However, a ResNet building block will learn the function \( y = \mathcal{F}(x) + id(x) \) where \( id(x) \) is a identity function \( id(x) = x \). The additional element \( id(x) \) is the key factor that improves the information flow through layers during training ResNets.

#### 3.2.2 Network design

A deep ResNet can be constructed from multiple ResNet building blocks that are serially connected to each other. To search for the best recognition performance, we suggest different configurations of ResNet with 20, 32, 44, 56, and 110 layers. For further details, see Appendix. These networks are denoted as ResNet-20, ResNet-32, ResNet-44, ResNet-56, and ResNet-110, respectively. In our implementations, a ResNet building block performs the learning of a function \( y = \mathcal{F}(x) + id(x) \) where \( id(x) = x \) and \( \mathcal{F}(x) \) is implemented by a sequence of layers: \( \text{Conv-BN-ReLU-Dropout-Conv-BN} \). Specifically, each ResNet block uses the convolutional layers (\text{Conv}) with \( 3 \times 3 \) filters. The Batch Normalization (\text{BN}) [59] and non-linear activation function \text{ReLU} [60] are applied after each \text{Conv}. To reduce possible overfitting, we add a Dropout layer [61] with a rate of 0.5 into each ResNet block, located between two \text{Conv} layers and after \text{ReLU}. Finally, another \text{ReLU} layer [60] is used after the element-wise addition. The proposed ResNet unit is shown in Figure 7. We refer the interested reader to the Supplementary Materials to see details of the proposed network architectures.

In this learning framework, all networks are designed for accepting images with size \( 32 \times 32 \) pixels and trained in an end-to-end manner using Stochastic Gradient Descent (SGD) algorithm [62] from scratch. The last fully-connected layer of the network represents the action class scores and its size can be changed corresponding to the number of action classes.

#### 3.2.3 Learning to recognize actions from skeleton-based representations

In order to recognize an action from a given skeleton sequence \( S \) with \( N \) frames \( \{F_1, F_2, ..., F_N\} \), we firstly encode all frames of \( S \) into a single RGB image \( I_{\text{RGB}} \) as mentioned in section 3.1.

\[ I_{\text{RGB}} = [F_1, F_2, ..., F_N], \]

where all elements of \( F_i \) for \( i \in [1, N] \) were normalized to the range of \([0, 255]\) as color pixels and rearranged according to their physical
More details about the action classes in each subset can be found in Table 1. The proposed ResNet building block. Here, the symbol $+$ denotes element-wise addition.

structure (see Figure 4). We then propose the use of ResNets to learn and classify the obtained color images. During the training phase, we minimize the cross-entropy loss function $\mathcal{L}_\mathcal{X}(y, \hat{y})$ between the action labels $y$ of $I_{\text{RGB}}$ and the predicted action labels $\hat{y}$ by the network over the training set $\mathcal{X}$. In other words, the network will be trained to solve the following problem

$$
\text{Arg} \min_{W} \mathcal{L}_\mathcal{X}(y, \hat{y}) = \text{Arg} \min_{W} \frac{1}{M} \sum_{i=1}^{M} \sum_{j=1}^{C} y_{ij} \log \hat{y}_{ij},
$$

where $W$ is the set of weights that will be learned by the network, $M$ denotes the number of samples in training set $\mathcal{X}$ and $C$ is the number of action classes.

4 Experiments

In this section, we evaluate the effectiveness of the proposed method on two benchmark datasets: MSR Action3D [36] and NTU-RGB+D [37]. To have a fair comparison with state-of-the-art approaches in the literature, we follow the evaluation protocols as provided in the original papers [36, 37]. Recognition performance will be measured by average classification accuracy.

4.1 Datasets and experimental protocols

4.1.1 MSR Action3D Dataset [36]: This dataset was collected with the Microsoft Kinect v1 sensor. It contains 20 different actions. Each action was performed by 10 subjects for two or three times. The experiments were conducted on 557 valid sequences of the dataset after removing the defective sequences. We followed the same experimental protocol as many other authors, in which the whole data is divided into three subsets named as AS1, AS2, and AS3. For each subset, five subjects (with IDs: 1, 3, 5, 7, 9) are selected for training and the rest (with IDs: 2, 4, 6, 8, 10) for test. More details about the action classes in each subset can be found in Table 1.

![Table 1](http://rose1.ntu.edu.sg/Datasets/)

4.1.2 NTU-RGB+D Dataset [37]: The NTU-RGB+D dataset is a very large-scale RGB+D dataset. To the best of our knowledge, the NTU-RGB+D is currently the largest dataset that contains skeletal data for HAR. It is a very challenging dataset due to the large intra-class variations and multiple viewpoints. Specifically, the NTU-RGB+D provides more than 56,000 videos, collected from 40 subjects for 60 action classes. The list of action classes in NTU RGB+D dataset includes: drinking, eating, brushing teeth, brushing hair, dropping, picking up, throwing, sitting down, standing up (from sitting position), clapping, reading, writing, tearing up paper, wearing jacket, taking off jacket, wearing a shoe, taking off a shoe, wearing on glasses, taking off glasses, putting on a hat/cap, taking off a hat/cap, cheering up, hand waving, kicking something, reaching into self pocket, hopping, jumping up, making/answering a phone call, playing with phone, typing, pointing to something, taking selfie, checking time (on watch), rubbing two hands together, bowing, shaking head, wiping face, saluting, putting palms together, crossing hands in front, sneezing/coughing, staggering, falling down, touching head (headache), touching chest (stomachache/heart pain), touching back (back-pain), touching neck (neck-ache), vomiting, fanning self, punching/shopping other person, kicking other person, pushing other person, putting other’s back, pointing to the other person, hugging, giving something to other person, touching other person’s pocket, handshaking, walking towards each other, and walking apart from each other.

The NTU RGB+D data was collected by the Microsoft Kinect v2 sensor. Each skeleton frame provides the 3D coordinates of 25 body joints. Two different evaluation criteria have been suggested, including Cross-Subject and Cross-View. In the Cross-Subject settings, 20 subjects (with IDs: 1, 2, 4, 5, 8, 9, 13, 14, 15, 16, 17, 18, 19, 25, 27, 28, 31, 34, 35, and 38) are used for training and the rest are used for test. For Cross-View settings, the sequences provided by cameras 2 and 3 are used for training and sequences provided by camera 1 are used for test.

4.2 Data augmentation

Very deep neural networks as ResNets require a lot of data to optimize. D-CNNs cannot learn and classify images effectively when training with an insufficient data amount. Unfortunately, we have only 557 skeleton sequences (less than 30 images per class) on MSR Action3D dataset [36]. Therefore, data augmentation has been applied to prevent ResNets from overfitting. We used random cropping, flip horizontally and vertically techniques to generate more training samples. More specifically, $8 \times 8$ cropping has been applied on $40 \times 40$ images to create $32 \times 32$ images. Then, their horizontally and vertically flipped images are also created. Figure 8 illustrates some data augmentation techniques on an image sample obtained from the MSR Action3D dataset [36]. For the NTU-RGB+D dataset [37], we do not apply any data augmentation method due to its very large-scale.
4.3 Implementation details

The image-based representations are computed directly from the original skeletons without any data preprocessing techniques, e.g., transforming the coordinate system, using a fixed number of frames, or removing noise. In this project, the MatConvNet toolbox [63] was used to design and train our deep learning networks. During the training phase, we use mini-batches of 128 images for ResNet-20, ResNet-32, ResNet-44, and ResNet-56 networks. For ResNet-110 network, we use mini-batches of 64 images. We initialize the weights randomly and train all networks for 120 epochs on a computer using GeForce GTX 1080 Ti GPU with 11GB RAM from scratch. The initial learning rate is set to 0.01 and is decreased to 0.001 at epoch 75. The last 45 epochs use a learning rate of 0.0001. The weight decay is set at 0.0001 and the momentum is 0.9.

5 Experimental Results and Analysis

This section reports our experimental results on the MSR Action3D dataset [36] and the NTU-RGB+D dataset [37]. We compare the obtained classification rates with state-of-the-art approaches in the literature [27, 36, 37, 41, 44, 44, 64–72, 72–85]. In addition, we also provide a detailed analysis about the computational efficiency of the proposed learning framework.

5.1 Result on MSR Action3D dataset

Experimental results on MSR Action3D dataset [36] are shown in Table 2. We achieved the best classification accuracy with the ResNet-32 model. More specifically, classification accuracies are highlighted in bold.

5.2 Result on NTU-RGB+D dataset

As shown in Table 4, the proposed learning framework reaches competitive results with an accuracy of 73.40% on Cross-Subject evaluation and an accuracy of 80.40% on Cross-View evaluation. The obtained results indicate that our method can deal with large intra-class variations and multiple viewpoints dataset as NTU-RGB+D [37]. Table 5 provides a comparison with published studies [36, 64–72, 74, 75, 79, 88]. It is clear that our method surpasses many previous approaches in the same experimental conditions. The learning curves of all ResNet configurations on two evaluation settings are shown in Figure 9d and Figure 9e.

5.3 Convergence rate analysis

Figure 9 shows the learning curves of the five proposed deep learning networks on the MSR Action3D [36] and the NTU-RGB+D [37] datasets. We can find that the training convergence rate of the networks are different on the two datasets. More specifically, on MSR
Action3D dataset [36], the proposed networks exhibit rapid convergence after 50 epochs at the beginning of the training process. Meanwhile, these networks can only start to converge after near 80 epochs on the NTU-RGB+D dataset [37]. This phenomenon can be explained by the complexity of feature spaces generated by the two datasets. With more than 56,000 videos collected from 40 subjects for 60 action classes, it is clear that the NTU-RGB+D dataset [37] is more complex than the MSR Action3D dataset [36]. This leads to deep learning networks needing more iterations to start convergence.

As shown in Table 2 and Table 4, the best recognition accuracy and configuration are highlighted in bold. More specifically, our experimental result on the MSR Action3D indicated that the baseline ResNet-32 achieved the best overall accuracy (99.53%). On the NTU RGB+D dataset, the ResNet-32 network was the best version on the Cross-View setting (80.40%). However, it worked worse than the ResNet-20 on the Cross-Subject setting. This is a weak aspect of the proposed approach. This limitation could be overcome by using ensemble learning techniques [90].

5.4 Analysis of training and prediction time

We take the AS1 subset of the MSR Action3D dataset [36] and the proposed ResNet-32 network for illustrating the computational efficiency of our deep learning framework. As shown in Figure 10, the proposed framework contains three main stages, including the encoding process from skeleton sequences into RGB images (Stage A), the supervised training stage (Stage B), and the prediction stage (Stage C). With the implementation in Matlab using MatconvNet toolbox [63]* on a single NVIDIA GeForce GTX 1080 Ti GPU†, without parallel processing, we take on average $4.15 \times 10^{-3}$ s per sequence during training stage. While the prediction time, including the time for encoding skeletons into RGB images and classification by the pre-trained ResNet, takes on average $21.84 \times 10^{-3}$ s per sequence (see Table 6). These results verify the effectiveness of our proposed method, not only in terms of accuracy but also in terms of computational efficiency.

Fig. 10: Three main stages of the proposed deep learning framework.

| Stage | Average processing time |
|-------|------------------------|
| Stage A | $20.8 \times 10^{-3}$ s per sequence (CPU time) |
| Stage B | $4.15 \times 10^{-3}$ s per sequence (GPU time) |
| Stage C | $21.84 \times 10^{-3}$ s per sequence (CPU time) |

Table 6: Execution time of each component of the proposed learning framework.

6 Conclusion and future work

In this paper, we have proposed a new 3D motion representation and an end-to-end deep learning framework based on ResNets [34] for human action recognition from skeleton sequences. To this end, we transformed the 3D joint coordinates carried in skeleton sequences...
into RGB images via a color encoding process. We then designed and trained different D-CNNs based on the ResNet architecture for learning the spatial and temporal dynamics of human motion from image-coded representations. The experimental results on two well-established datasets, including the largest RGB-D dataset currently available, demonstrated that our method can achieve state-of-the-art performance whilst requiring a low computation cost for the training and prediction stages. For future work, we aim to extend this research by investigating a new skeletal encoding method, in which the joint Euclidean distances and orientation relations between joints will be exploited. In addition, we plan to build a better feature learning and classification framework by experimenting some new and potential architectures based on the idea of ResNet such as Inception Residual Networks (Inception-ResNet) [91] or Densely Connected Convolutional Networks (DenseNet [92]). In order to overcome the limitations of Kinect sensors when dealing with HAR in which the lighting conditions are available, we will also consider some new approaches for estimating human body key-points such as deep learning based approaches [93, 94], KWE and KCE methods [10].

We hope that our study will open a new door for the computer vision learning based approaches [93, 94], KWE and KCE methods [10].

7 References

1. S. Ransanihe, F. Al Machot, H. C. Mayr. “A review on applications of activity recognition systems with regard to performance and evaluation”, International Journal of Distributed Sensor Networks, 2015, 12, 20150028, 2015.
2. W. Niu, J. Long, D. Han, Y.-W. Fan. “Human activity detection and recognition for video surveillance”, IEEE International Conference on Multimedia & Expo (ICME), volume 1, pp. 1–4, 2004.
3. W. Lin, M.-T. Sun, R. Poovorawan, Z. Zhang. “Human activity recognition for video surveillance”, IEEE International Symposium on Circuits and Systems (ISCAS), pp. 2737–2740, 2008.
4. C. A. Pickinger, K. J. Burnham, M. J. Richardson. “A research study of hand gesture recognition technologies and applications for human vehicle interaction”, Institution of Engineering and Technology Conference on Automotive Electronics, pp. 1–5, 2007.
5. P. Sonwalkar, T. Sakhare, A. Patil, S. Kale. “Hand gesture recognition for real time human machine interaction system”, International Journal of Engineering Trends and Technology (IJETT), 2013, 2, 262–264, 2015.
6. J. Pantisi, D. Stoyanov, D. McIvor, B. L. Go, Y. Guang. “Ambient and wearable sensor fusion for activity recognition in healthcare monitoring systems”, International workshop on wearable and implantable body sensor networks (BSN 2007), pp. 208–212, 2007.
7. R. Poppe. “A survey on vision-based human action recognition”, Image and Vision Computing, 2008, 2008, 26, 976–990, 2010.
8. A. Akula, A. K. Shah, R. Ghosh. “Deep learning for human activity recognition in infrared images”, Cognitive Systems Research, 2015, 30, 146–154, 2016.
9. S. Z. Gurzhi, W. L. Melvin, D. B. Williams. “Detection and identification of human targets in radar data”, Signal Processing, Sensor Fusion, and Target Recognition XVI, volume 8665, pp. 86650F, 2013.
10. G. Albazadeh. “A new statistical-based kurstos wavelet energy feature for texture classification of human faces”, IEEE Trans. Geoscience and Remote Sensing, 2014, 52, 3438–3447, 2016.
11. Z. Tizhand, G. Albazadeh. “Unsupervised texture-based sar image segmentation using spectral regression and gabor filter bank”, Journal of the Indian Society of Remote Sensing, 2015, 43, 177–186, 2016.
12. J. Han, L. Shao, D. Xu, J. Shotton. “Enhanced Computer Vision With Microsoft Kinect Sensor: A Review”, IEEE Transactions on Cybernetics, 2013, 43, 1318–1324, 2012.
13. “ASUS Xtion PRO”, https://www.asus.com/3D-Sensor/Xtion_PRO/, accessed: 2010-09-30.
14. “Intel RealSense”, https://www.intel.com/content/www/us/en/architecture-and-technology/realsense-overview.html, accessed: 2010-09-30.
15. J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Cook, R. Moore. “Real-time human pose recognition in parts from single depth images”, Communications of the ACM, 2011, 54, 116–124, 2013.
16. J. Wang, Z. Liu, Y. Wu, J. Yuan. “Mining actionlet ensemble for action recognition with depth cameras”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1290–1297, 2012.
17. L. Xia, C.-C. Chen, J. K. Aggarwal. “View invariant human action recognition using histograms of 3D joints”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR), pp. 20–27, 2012.
18. R. Chaudhry, F. Olli, G. Kurillo, R. Bajcsy, R. Vidal. “Bio-inspired Dynamic 3D Discriminative Skeletal Features for Human Action Recognition”, IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR), pp. 471–478, 2013.
19. R. Vemula, Y. T. F. Arrate, R. Chellappa. “Human Action Recognition by Representing 3D Skeletons as Points in a Lie Group”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 558–595, 2014.
20. W. Ding, K. Liu, X. Fu, F. Feng. “Profile HMVs for skeleton-based human action recognition”, Signal and Image Communication Processing and Applications, 2015.
21. A. Krizhevsky, I. Sutskever, G. E. Hinton. “Imagenet classification with deep convolutional neural networks”, Advances in Neural Information Processing Systems (NIPS), pp. 1097–1105, 2012.
22. A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, L. Fei-Fei. “Large-scale video classification with convolutional neural networks”, Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR), pp. 1725–1732, 2014.
23. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna. “Rethinking the Inception Architecture for Computer Vision”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2818–2826, 2016.
24. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich. “Going Deeper with Convolutions”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–9, 2015.
25. M. Telgarsky. “Benefits of depth in neural networks”, arXiv preprint arXiv:1602.04485, 2016.
26. K. He, X. Zhang, S. Ren, J. Sun. “Deep residual learning for image recognition”, Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.
27. H. H. Pham, L. Khoudour, A. Crouzil, P. Zeegers, S. A. Velastin. “Learning and recognizing human action from skeleton movement with deep residual networks”, 8th International Conference of Pattern Recognition Systems (ICPRS 2017), pp. 1–6, 2017.
28. W. Li, Z. Zhang, Z. Liu. “Action recognition based on a bag of 3D points”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 9–14, 2010.
29. A. Shahroudy, J. Liu, T.-N. Ng, G. Wang. “NTU RGB+D: A large scale dataset for 3D human activity analysis”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1010–1019, 2016.
30. B.-J. Yoon. “Hidden Markov models and their applications in biological sequence analysis”, Current genomics, 2006, 7, 402–415, 2006.
31. F. Lv, R. Nevatia. “Recognition and segmentation of 3D human action using HMM and multi-class adaboost”, European Conference on Computer Vision (ECCV), pp. 359–372, 2006.
32. D. Wu, L. Shau. “Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 724–731, 2014.
33. I. Lou, W. Wang, H. Q. Cao. “Shape and geometric constrained dictionary learning for action recognition from depth maps”, Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 1809–1816, 2013.
34. S. Sangyoon, J. Schmidhuber. “Long Short-Term Memory”, Neural Computation, 1997.
35. A. Graves. “Supervised sequence labelling with recurrent neural networks”, Studies in Computational Intelligence, 2008.
36. Y. Da, W. Wang, L. Wang. “A hierarchical recurrent neural network for skeleton based action recognition”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1110–1118, 2015.
of Visual Communication and Image Representation, 25(1), pp. 12–23, (2014).

Z. Gao, J. m. Song, H. Zhang, A.-A. Liu, Y.-S. Xiao, G. p. Xu. “Human action recognition via multi-modality information”, Journal of Electrical Engineering and Technology, 9(2), pp. 739–748, (2014).

A. W. Vieira, E. R. Nascimento, G. E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov. ”Reducing internal covariate shift”, International Conference on Machine Learning, (2015), pp. 358–367,

Advances in Multimedia Information Processing – PCM 2017, (2020).

R. Vemulapalli, R. Chellappa. "Rolling Rotations for Recognizing Human Actions from 3D Skeletal Data”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4471–4479, (2016).

Z. Luo, B. Peng, D.-A. Huang, A. Alahi, F. Li. "Unsupervised Learning of Long-Term Motion Dynamics”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7101–7110, (2017).

X. Li, Z. Zhang. “Improved key pose model for skeleton-based action recognition”, B. Zeng, Q. Huang, A. El Saddik, H. Li, S. Jiang, X. Fan (Eds.), Advances in Multimedia Information Processing – PCM 2017, pp. 358–367, (Cham,2018).

D. C. Lavison, H. Tabia, D. Picard. “Learning features combination for human action recognition from skeleton sequences”, Pattern Recognition Letters, 99, pp. 13–20, (2017).

Y. Ibhai, S. Ibhai, L. Ballihi, M. Sadgal. “Skeletal-based human action recognition for elderly monitoring systems”, IET Computer Vision, 12(4), pp. 16–26, (2017).

M. Zanfir, M. Leordeanu, C. Sminchisescu. “The moving pose: An efficient 3d kinematics descriptor for low-latency action recognition and detection”, Proceed- ings of the IEEE international conference on computer vision, pp. 2752–2759, (2013).

T. G. Dietterich. “Ensemble methods in machine learning”, International workshop on multiple classifier systems, pp. 1–15, (Springer, 2000).

C. Szegedy, S. Ioffe, V. Vanhoucke, A. A. Alemi. “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning”, AAAI Conference on Artificial Intelligence, pp. 4278–4284, (2017).

G. Huang, Z. Liu, K. Q. Weinberger. “Densely Connected Convolutional Net- works”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2291–2299, (2017).

Z. Cao, T. Simon, S. E. Wei, Y. Sheikh. “Realtime Multi-Person 2D Pose Estima- tion using Part Affinity Fields”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2017).

I. K. Riza Alp Guler, Natalia Neverova. ”Densepose: Dense human pose estima- tion in the wild”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2018).

Appendix
This section describes the network architectures in detail. To build 20-layer, 32-layer, 44-layer, 56-layer, and 110-layer networks, we stack the proposed ResNet building units as following:

Baseline 20-layer ResNet architecture
3x3 Conv., 16 filters, BN, ReLU
Residual unit: Conv.-BN-ReLU-Dropout-Conv., 16 filters—> ReLU Residual unit: Conv.-BN-ReLU-Dropout-Conv., 16 filters—> ReLU Residual unit: Conv.-BN-ReLU-Dropout-Conv., 16 filters—> ReLU Residual unit: Conv.-BN-ReLU-Dropout-Conv., 32 filters—> ReLU
**Baseline 32-layer ResNet architecture**

| Layer Configuration | Filter Configuration | Activation Functions |
|---------------------|---------------------|---------------------|
| 3x3 Conv., 16 filters, BN, ReLU | 32 filters | ReLU |
| Residual unit: Conv.-BN-ReLU-Dropout-Conv.,16 filters-+ | ReLU |
| Residual unit: Conv.-BN-ReLU-Dropout-Conv.,16 filters-+ | ReLU |
| Residual unit: Conv.-BN-ReLU-Dropout-Conv.,16 filters-+ | ReLU |
| Global mean pooling | |
| FC layer with \(n\) units where \(n\) is equal the number of action class. | Softmax layer |

**Baseline 44-layer ResNet architecture**

| Layer Configuration | Filter Configuration | Activation Functions |
|---------------------|---------------------|---------------------|
| 3x3 Conv., 16 filters, BN, ReLU | 32 filters | ReLU |
| Residual unit: Conv.-BN-ReLU-Dropout-Conv.,16 filters-+ | ReLU |
| Residual unit: Conv.-BN-ReLU-Dropout-Conv.,16 filters-+ | ReLU |
| Residual unit: Conv.-BN-ReLU-Dropout-Conv.,16 filters-+ | ReLU |
| Global mean pooling | |
| FC layer with \(n\) units where \(n\) is equal the number of action class. | Softmax layer |

**Baseline 56-layer ResNet architecture**

| Layer Configuration | Filter Configuration | Activation Functions |
|---------------------|---------------------|---------------------|
| 3x3 Conv., 16 filters, BN, ReLU | 32 filters | ReLU |
| Residual unit: Conv.-BN-ReLU-Dropout-Conv.,16 filters-+ | ReLU |
| Residual unit: Conv.-BN-ReLU-Dropout-Conv.,16 filters-+ | ReLU |
| Residual unit: Conv.-BN-ReLU-Dropout-Conv.,16 filters-+ | ReLU |
| Global mean pooling | |
| FC layer with \(n\) units where \(n\) is equal the number of action class. | Softmax layer |
Baseline 110-layer ResNet architecture

3x3 Conv., 16 filters, BN, ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Residual unit: Conv.-ReLU-Dropout-Conv., 16 filters + ReLU
Global mean pooling

FC layer with n units, where n is equal the number of action class.
Softmax layer