AIMS Mathematics, 9(7): 17901–17916.
DOI: 10.3934/math.2024871
Received: 13 December 2023
Revised: 24 March 2024
Accepted: 28 April 2024
Published: 24 May 2024

http://www.aimspress.com/journal/Math

Research article

Enhancing skeleton-based human motion recognition with Lie algebra and memristor-augmented LSTM and CNN

Zhencheng Fan[1,*], Zheng Yan[1], Yuting Cao[2], Yin Yang[2] and Shiping Wen[1]

[1] Australian AI Institute, Faculty of Engineering and Information Technology, University of Technology Sydney, NSW 2007, Australia

[2] College of Science and Engineering, Hamad Bin Khalifa University, 5855, Doha, Qatar

* **Correspondence:** Email: zhencheng.fan@student.uts.edu.au.

Abstract: Lately, as a subset of human-centric studies, vision-oriented human action recognition has emerged as a pivotal research area, given its broad applicability in fields like healthcare, video surveillance, autonomous driving, sports, and education. This brief applies Lie algebra and standard bone length data to represent human skeleton data. A multi-layer long short-term memory (LSTM) recurrent neural network and convolutional neural network (CNN) are applied for human motion recognition. Finally, the trained network weights are converted into the crossbar-based memristor circuit, which can accelerate the network inference, reduce energy consumption, and obtain an excellent computing performance.

Keywords: human motion recognition; Lie algebre; memristor; LSTM; neural network
Mathematics Subject Classification: 68T07, 68T10

1. Introduction

As artificial intelligence evolves, particularly with advancements in deep learning neural networks, human action recognition has found extensive applications in healthcare [1–3]. Due to its wide application in video surveillance, autonomous driving, physical education, and other fields, it can greatly improve people's quality of life and simplify work processes [4]. In the realm of human action recognition, visual approaches to represent human actions can be broadly categorized into three groups: RGB-oriented [5], skeleton-driven, and those rooted in depth maps [6]. Among them, bone-based (skeleton-based) representations have received extensive attention due to their viewpoint independence and ease of describing motion.

At present, the data collection technology of human body posture is becoming more and more mature. There are 3D bone data acquisition devices such as Kinect on the hardware and Openpose [7],

which uses deep neural networks for bone recognition and synthesis on the software level. Human skeletal data can be seamlessly extracted from either videos or images. In the context of human action recognition, the manner in which actions are represented is pivotal. An optimal representation should precisely encapsulate the spatial dynamics associated with joints and bones. Predominantly, unit quaternions and Euler angles serve as the go-to methodologies to encapsulate human movement. However, the unit quaternion may cause numerical and analytical difficulties, and the Euler angle will have the problem of a Vientiane lock. Representation methods based on Lie groups and Lie algebras [8] solve these problems well and provide a more reasonable representation method for the representation of human behavior. At the same time, using Lie algebra to represent bone data can not only gain computational advantages, but can also be combined with standard bone data, ignoring the effect of bone length. At the same time, the representation method based on Lie algebra divides the human skeleton data into five parts; we can calculate these five parts separately, and further realize the accurate judgment of an action.

At the same time, many model methods and target detection algorithms have also been proposed for human action recognition in deep learning, including energy-relation diagrams (ERD), 3 layers of long short-term memory (LSTM-3LR) [9], stacked recurrent network (SRNN) [10], you only look once (YOLO) [11], etc. These methods have made important contributions to human action recognition and target detection. However, an effective model often needs to combine data of multiple dimensions for calculations, and contains a large number of parameters, which requires a lot of computing power, and may consume a lot of time when processing skeleton data, which makes the model less applicable. Therefore, the model cannot achieve a good performance when dealing with real-time streaming information.

To address these issues, the advent of memristors has significantly contributed to accelerating model computation as one of the branches of neuromorphic computing. The emergence of memristors provides options for the hardware implementation of neuromorphic computing. Memristor crossbar-based networks can achieve extremely high parallel speeds and consumes very little energy during computation. This has comprehensive practical value [12].

This brief applies Lie algebra and standard bone length data to represent human skeleton data. A multi-layer LSTM recurrent neural network and convolutional neural network (CNN) are applied for human motion recognition. Finally, the trained network weights are converted into the crossbar-based memristor circuit, which can accelerate the network inference, reduce energy consumption, and obtain an excellent computing performance.

Our work advances human action recognition and neuromorphic computing with key contributions: (1) Implemented network structures with memristors, demonstrating minimal accuracy loss, and showcasing the efficiency of memristor technology in deep learning; (2) adapted the use of Lie algebra for skeletal representation within a memristor-based network structure for the first time, enhancing the integration of advanced motion capture techniques with neuromorphic computing; (3) and explored potential applications of memristors in neuromorphic computing, setting a foundation for future low-power, high-speed computing solutions.

2. Proposed method

2.1. Skeletal human motion representation

As a fundamental problem in human action-related tasks, there are currently three popular methods: RGB-based, depth-image-based, and bone-based methods. Considering generality and easy availability, this paper chooses a representation method based on skeletal data [13].

For a more refined representation of human action data, while mitigating the effect of skeletal length variation, we've adopted the methodologies of Lie algebra and the standard skeleton data representation. Lie algebras are mathematical structures essential in studying continuous symmetry and differential equations. They serve as the algebraic counterparts to Lie groups, which represent continuous transformations. A Lie algebra is defined as a vector space equipped with a binary operation known as the Lie bracket, adhering to bilinearity and the Jacobi identity. These properties ensure that the algebra captures the concept of infinitesimal symmetries. Lie algebras play a crucial role in mathematics and physics, especially in differential equations, geometry, and quantum mechanics. As visualized in Figure 1, we configure the local coordinate framework for e_m by imposing minimal rotation and translation adjustments to the global coordinate setup. This results in e_m serving as the definitive reference for the x-axis's position and orientation, taking its starting joints as the coordinate's inception point. Post this transformation, the relative positioning of e_n within the localized system of e_m is discerned, signified as e_n^m. Subsequent to this, we engineer a 3D rigid transformation articulated as $\begin{pmatrix} R_{n,m} & d_{n,m} \\ 0 & 1 \end{pmatrix}$, where $R_{n,m}$ constitutes a 3×3 rotational matrix, and $d_{n,m}$ is the corresponding 3D translational vector, facilitating the shift of e_m to align with the position and orientation of e_n.

$$\begin{bmatrix} e^m_{n,end} \\ 0 \end{bmatrix} = \begin{bmatrix} R_{n,m} & d_{n,m} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} l_n \\ 0 \\ 0 \\ 1 \end{bmatrix}. \tag{2.1}$$

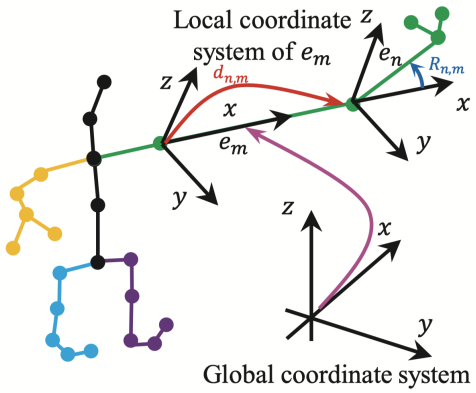

Figure 1. Lie group depiction of skeletal translation and rotation.

In the context of this representation, $e^m_{n,end}$ signifies the terminal joint of e_n^m, and l_n denotes the length of e_n. Analogously, by employing a distinct transformation matrix, we ascertain e_m's position

within the local system of e_n. Consequently, given M as the total count of bones, we derive $M \times (M-1)$ transformation matrices. From a computational standpoint, a 3D rigid transformation can be characterized within the framework of the special Euclidean group, denoted as $SE(3)$. Ultimately, a skeleton can be characterized as a trajectory in the multi-dimensional space $SE(3) \times ... \times SE(3)$.

To negate the impact of varying bone lengths, which essentially eliminates the influence of diverse body types on identical posture evaluations, we adopt a standard-length bone data for classification. This implies that, only the rotation matrix becomes essential for a human pose representation. Moreover, given that the human form can be depicted as a linkage structure with five primary segments — the spine, a pair of legs, and a pair of arms, as illustrated in Figure 2 — our focus is on computing the rotation matrix specifically between two contiguous bones sharing a joint, rather than between any arbitrary bones within a segment. This approach retains the inherent structure of the skeletal framework by honoring the anatomical constraints between chains. A subsequent advantage of this methodology is the reduced count of rotation matrices, thus offering potential computational efficiencies.

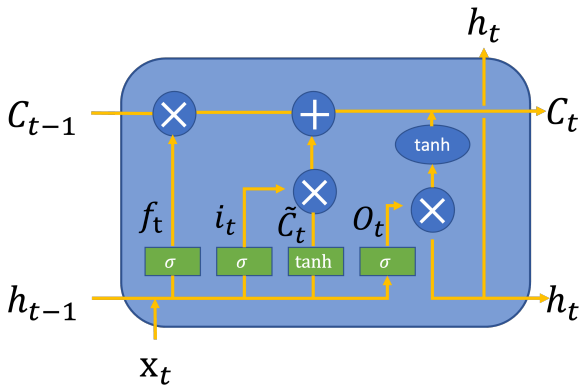

Figure 2. Schema of LSTM unit.

Operationally, the axis-angle representation $(\mathbf{n}, \theta)$ is initially derived as follows:

$$\mathbf{n} = \frac{cross(e_n, e_m)}{\|cross(e_n, e_m)\|}, \tag{2.2}$$

$$\theta = arccos(e_n, e_m). \tag{2.3}$$

Here, *cross* signifies the outer product, and Δ represents the inner product. Following this, the rotation matrix $R_{n,m}$ is inferred via the Rodriguez formula:

$$R_{n,m} = I + sin(\theta)\mathbf{n}^{\wedge} + (1 - cos(\theta))\mathbf{n}^{\wedge\mathbf{2}}. \tag{2.4}$$

In our discussion, $I \in \mathbb{R}^{3\times3}$ represents the identity matrix, while $\mathbf{n}^{\wedge}$ signifies the skew-symmetric matrix associated with $\mathbf{n}$. It's essential to recognize that this collection of rotation matrices is a member of the special orthogonal group $SO(3)$. Consequently, the skeleton can be envisioned as navigating a path in $SO(3)\times...\times SO(3)$. Given the intricate nature of regression within the curved domain $SO(3)\times...\times SO(3)$, we aim to convert this domain to its tangent space, which is seen as the Lie algebra $SO(3)\times...\times SO(3)$. To achieve this, we employ an approximate logarithm map method:

$$\omega(R_{n,m}) = \frac{1}{2sin(\theta(R_{n,m}))} \begin{bmatrix} R_{n,m}(3,2) - R_{n,m}(2,3) \\ R_{n,m}(1,3) - R_{n,m}(3,1) \\ R_{n,m}(2,1) - R_{n,m}(1,2) \end{bmatrix}, \tag{2.5}$$

$$\theta(R_{n,m}) = arccos(\frac{Trace(R_{n,m}) - 1}{2}). \tag{2.6}$$

In essence, the skeleton is projected onto a set of Lie algebra vectors, denoted as follows: $\omega = [\omega_1^{1^T}, ..., \omega_{K_1}^{1^T}, ..., \omega_1^{C^T}, ..., \omega_{K_C}^{C^T}]^T$, where C indicates the total chains (for our setup, $C = 5$, which mirrors human movement) and $K_c(c \in 1, ..., C)$ signifies the number of bones in the c-th chain reduced by one.

The bone representation method we've employed offers dual benefits: first, it negates the effect of bone length on the final outcomes; and second, it curtails the computational parameter count needed for the subsequent neural network processing.

2.1.1. Utilizing LSTM and CNN architectures

In pursuit of an enhanced accuracy, this study integrates both LSTM and CNN architectures, as described by [14], to handle data presented in the Lie algebra form. Recognizing that human skeletal action data encompasses both temporal and spatial dimensions, an amalgamation of LSTM and CNN networks is deemed optimal. This is because the LSTM structure excels at amalgamating temporal context features, while CNN thrives at spatial feature extraction [15–17].

As an advanced version of the traditional recurrent neural network (RNN), LSTM proficiently captures long-range temporal characteristics. Importantly, it addresses the notorious gradient explosion or vanishing challenges encountered in conventional RNNs, marking it particularly adept for a time-series data analysis. A classic LSTM model is employed here. The concept of gating–encompassing the input, forget, and output gates lies at the core of the LSTM's functionality. The mathematical computations within the LSTM unit are articulated as follows:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_t), \tag{2.7}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \tag{2.8}$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C), \tag{2.9}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \tag{2.10}$$

$$C_t = f_t \otimes C_{t-1} + i_t \otimes \tilde{C}_t, \tag{2.11}$$

$$h_t = o_t \otimes tanh(C_t). \tag{2.12}$$

Here, $\otimes$ represents the Hadamard product, C_t and h_t indicate the cell and hidden states, respectively, and f_t, i_t, and o_t distinguish between the forget, input, and output gates, respectively. Within the scope of this research, a tri-layered LSTM architecture is leveraged to mine temporal features from the dataset.

As previously highlighted, the LSTM network excels at extracting features in the temporal domain. To further amplify our model's classification efficacy, we've incorporated auxiliary network structures as delineated in [14, 18]. AlexNet, which is a seminal deep CNN, has demonstrated a robust performance across various applied tasks. By integrating the capabilities of both the LSTM and AlexNet architectures, our approach is adept at capturing the nuanced interplay between the temporal and spatial features within the dataset. This synergy ensures that the strengths of one network offset any limitations of the other.

2.2. *Memristor-based LSTM and CNN*

Both LSTM and CNN networks have a very large amount of parameters, which makes practical applications difficult. In many edge computing devices, the computing power that meets the conditions cannot be provided. At the same time, for the future computing systems, power consumption and speed are two goals currently pursued. Our ultimate goal is to want low-power, fast computing devices. While neuromorphic computing has significant advantages, it can solve complex problems while consuming very little power and area [19]. This feature gives neuromorphic computing the ability to be widely used.

In recent years, the memristor [20] has received great attention as one of the directions of neuromorphic computing. Memristors, which are defined as memory resistors, are passive electronic components capable of retaining a voltage history, thus embodying a non-volatile memory function. This feature facilitates their application in neuromorphic computing systems by emulating synaptic connections. Unlike binary-operating transistors, memristors support analog computations through variable resistance values, enhancing the computing efficiency by enabling complex computations within memory units, thereby minimizing the data transfer between the processors and memory. This physical property, which can be tuned to a specific resistance value by applying a voltage to change its conductivity, is crucial for its functionality. Remarkably, this characteristic can be retained in the memristor even after a power down. By organizing memristors into a grid of crossbars [21, 22], many neural network computations can be performed in parallel, further leveraging the unique capabilities of memristors in neuromorphic computing architectures.

2.2.1. Memristor crossbar

As shown in Figure 3, a single layer feed forward neural network is implemented by using a 5×6 crossbar with four inputs and three outputs. Memristors are placed at the intersections of the bar structure and represent the weights of the network. Thanks to this special structure, the input can be processed in parallel, resulting in a faster speed [23]. Similarly, by leveraging the output from the prior crossbar layer as the input for the subsequent one, we can construct a multi-tiered feedforward neural network.

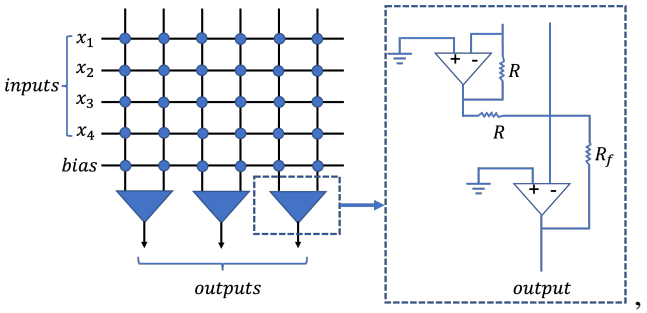

Figure 3. Schema of memristor crossbar.

2.2.2. Memristor-based LSTM

Artificial neural networks (ANNs) have become a cornerstone in the field of machine learning, mimicking the structure and function of the human brain's neural networks. These computational

models consist of nodes or neurons, organized in layers, that process input data through a series of transformations and connections. The most basic form of these networks includes fully connected layers, where each neuron in one layer connects to every neuron in the subsequent layer, thus facilitating the learning of complex patterns in the data.

The transition from theoretical neural network models to practical applications within computer systems has been marked by significant advancements in the computational power and algorithms. This evolution has enabled the implementation of complex neural network architectures, such as CNNs for image processing and RNNs for sequential data analyses. Among these, LSTM networks, a specialized form of RNNs, have been particularly effective in capturing long-term dependencies in the sequence data, which is a critical aspect in fields such as natural language processing and time series forecasting.

There are already many LSTM circuits implemented with memristors. A crossbar based LSTM architecture was proposed [24], and the effectiveness of the structure was demonstrated by a textual sentiment analysis. Then, an on-chip trained LSTM, namely the MbLSTM, was proposed in [25] . Similar to [24], the activation functions sigmoid and tanh were approximately implemented through intentionally designing circuit parameters. In this paper, in order to realize the LSTM network structure, we adopt the scheme in [25]. Instead, we ended up using ex-situ training to write the trained weights into the LSTM architecture.

According to the architecture in [25], the general structure of LSTM cell is shown in Figure 4. Thus we can get the following:

$$c^t(k) = -i^t(k) \cdot [-a^t(k)] - [-f^t(k) \cdot c^{t-1}(k)] \tag{2.13}$$

and

$$h^t(k) = -o^t(k) \cdot tanh(c^t(k)), \tag{2.14}$$

where tanh in (2.14) is the approximate activation function implemented by a circuit. Moreover, the multiplication is perfermed by existing analog multipliers. $h^t(k)$ is converted to $[-V_r, V_r]$ for the next step

$$V_h^t(k) = \frac{R_4}{R_3} h^t(k). \tag{2.15}$$

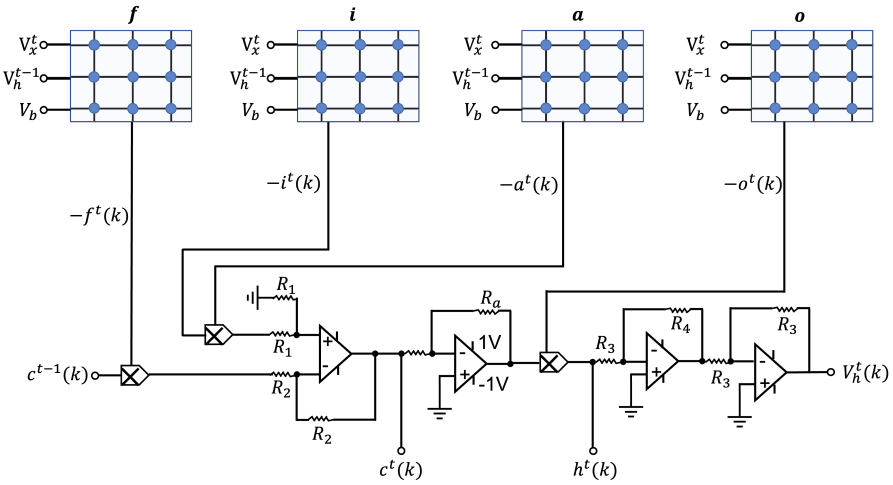

Figure 4. Memristor crossbar based LSTM cell, where f, i, a, o are four memristor based LSTM units.

2.2.3. Memristor-based CNN

As another auxiliary network of the overall network, CNNs can capture spatial features well. In [26], the authors proposed a simulated memristor crossbar implementation of the CNN. In this structure, the convolution of the image is not done once, but divided into multiple iterations. Thus, considering the size of the memristor crossbar, an image is divided into multiple inputs, and the final convolution results are spliced to obtain the final result. Certain arrangements were made through the convolution kernel to realize the CNN structure that processed the entire picture at one time in [27]. However in this structure, if the size of the picture increases, the number of memristors significantly increases, which is one of the drawbacks of this method. Meanwhile, a new convolution method was proposed to reduce the parameters by about 75% and reduce the number of multiplication computations for the convolutional layers by 30% within an acceptable accuracy loss [28]. A fully hardware-implemented memristor convolutional neural network was proposed in [29]. In this brief, we consider the possibility of a practical application and reduce the number of the memristor. We adopt the structure in [26] to implement the CNN. Figure 5 shows a single column of the memristor crossbar for performing convolution. Same as in [26], we set $V_{S1} = -1V, V_{D1} = 0V, V_{S2} = 0V, V_{D2} = 1V$, M_g is a memristor used to control the feedback gain, σ_β is the bias, and R_f is the unity gain. In this structure, the convolution kernels are determined in advance during the network training process. Since the convolution kernel may have negative values, in order to allow the convolution operation to process both positive and negative values, the convolution kernel and input values are divided into two column vectors:

$$\begin{bmatrix} 0.1 & -0.2 & 0.3 \\ -0.4 & 0.5 & -0.6 \\ 0.7 & -0.8 & 0.9 \end{bmatrix} \rightarrow \begin{bmatrix} 0.9 \\ -0.6 \\ 0.3 \\ -0.8 \\ 0.5 \\ -0.2 \\ 0.7 \\ -0.4 \\ 0.1 \end{bmatrix} \rightarrow \begin{bmatrix} 0.9 \\ 0 \\ 0.3 \\ 0 \\ 0.5 \\ 0 \\ 0.7 \\ 0 \\ 0.1 \end{bmatrix} \begin{bmatrix} 0 \\ 0.6 \\ 0 \\ 0.8 \\ 0 \\ 0.2 \\ 0 \\ 0.4 \\ 0 \end{bmatrix}. \tag{2.16}$$

As shown in (2.16), a convolution kernel will be rearranged into two column vectors, each storing the absolute value of the original value of the convolution kernel. One column represents positive values and the other column represents negative values:

$$\begin{bmatrix} 0 & 0.5 & 0.3 \\ 0.5 & 0.8 & 0.5 \\ 0 & 0.5 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 \\ 0.5 \\ 0.3 \\ 0.5 \\ 0.8 \\ 0.5 \\ 0 \\ 0.5 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 \\ 0.5 \\ 0.3 \\ 0.5 \\ 0.8 \\ 0.5 \\ 0 \\ 0.5 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ -0.5 \\ -0.3 \\ -0.5 \\ -0.8 \\ -0.5 \\ 0 \\ -0.5 \\ 0 \end{bmatrix}. \tag{2.17}$$

As shown in (2.17), for input x, the permutation method is different. x is divided into two columns, each containing all the values in x, one positive and another negative.

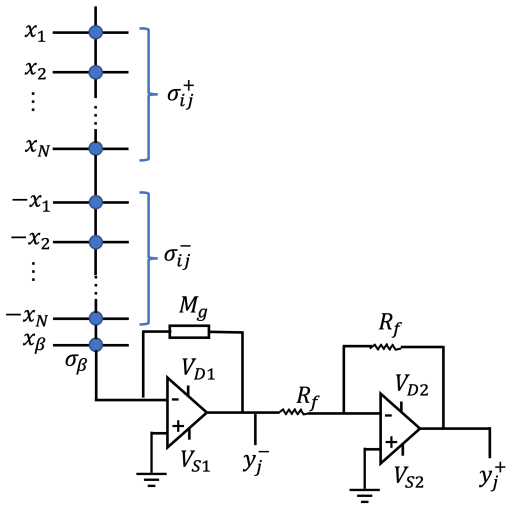

Figure 5. A single column of memristor crossbar for performing convolution.

Finally, through (2.18), the convolution kernels are converted to conductivity values:

$$\sigma^{\pm} = \frac{(\sigma_{max} - \sigma_{min})}{max(|W|)} W^{\pm} + \sigma_{min}. \tag{2.18}$$

For the final output activation function sigmoid, a circuit simulation is also used here to approximate the sigmoid function [24–26]. At this point, we implement a single convolution operation. Based on the convolution operation under this structure, it is impossible to process all input values at one time; therefore, it is necessary to divide a feature map into multiple inputs, then ultimately splicing to obtain the result of the convolution operation. To some extent, this approach reduces the space of the memristor, sacrificing a certain amount of time.

2.2.4. Dataset

In this brief, we use H3.6M dataset [30], which contains 3.6 million 3D skeleton data of human action sequences, and the NTU RGB+D dataset [31], which is a comprehensive collection encompassing 56,578 samples of 60 distinct action categories. These actions are captured from multiple perspectives, including a frontal view, two lateral views, and oblique views at 45 degrees to the left and right. The dataset features performances by 40 participants, whose ages range from 10 to 35 years, providing a diverse basis for action recognition research. According to the method from [13,32], we transformed the 3D data into Lie algebras; in order to exclude the effect of bone length on the classification, we used a uniform standard bone length.

2.3. System structure overview

The architecture of our system is depicted in Figure 6. Initially, the skeleton data from the dataset is transformed into the Lie algebra representation. This approach diverges from traditional methods by utilizing skeletal data encoded in the Lie algebra, as opposed to the direct use of the skeleton data.

Inspired by the methodologies in [13, 14, 33, 34], we compute temporal-domain features (TPF) from the transformed data. A key modification in our process is reshaping the Lie algebra-encoded skeletal data to align with the TPF extraction techniques described in these references, ensuring our method remains consistent with established practices. Unlike, [14] where the LSTM network inputs were spatial-domain features (SPF), our model's inputs are action sequences transformed into a Lie algebra.

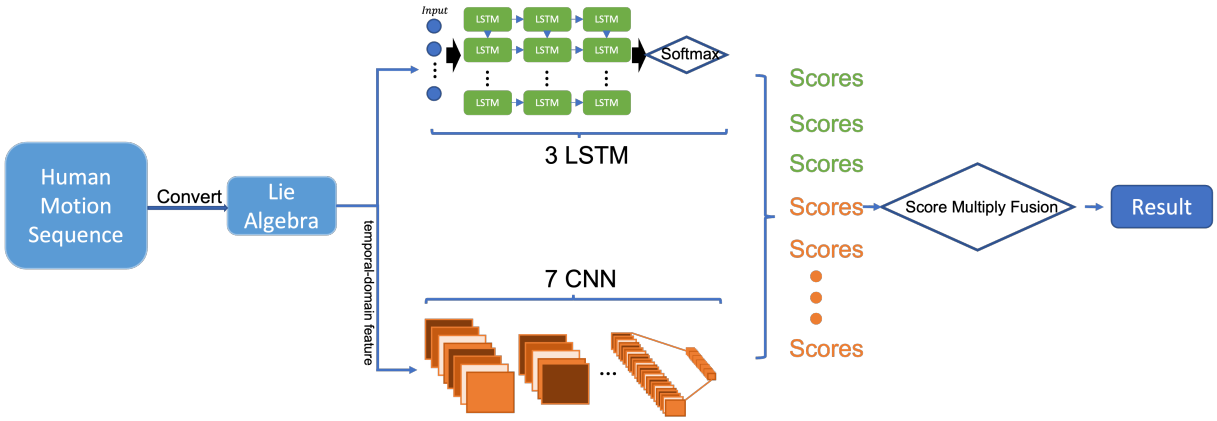

Figure 6. Overview of the proposed system.

In our model, frame indices are denoted by $i \in (1, ..., T)$ and elements within the Lie algebra vector ω by $j \in (1, ..., K)$, where $K = \sum_{c=1}^{C} K_c$. For simplicity, we refer to elements in ω as bones. Our three-layer LSTM architecture processes this data in stages: the first layer captures the overall motion information from the bones represented in Lie algebra; the second layer employs a dedicated LSTM to model the spine; and the final layer uses another set of LSTMs to analyze the remaining skeletal parts.

For computing the final output score of each network, we adopt a multiply-score fusion method as described in (2.19):

$$label = Fin(max(v_1 \circ v_2 \cdots v_9 \circ v_{10})). \tag{2.19}$$

In this context, v represents the score vector, with $\circ$ signifying element-wise multiplication. Meanwhile, Fin identifies the index corresponding to the maximum element.

First, we train the weights of the network via software network, and then map the weights to the memristor circuit through a transformation. The resulting circuit achieves a significant improvement in the inference speed over the software-implemented network.

2.4. Experiment and result

The Human 3.6M dataset contains 3.6 million 3D human pose data, including 17 scenes: discussion, smoking, taking pictures, talking on the phone, and so on. First, we convert the dataset to a Lie algebra representation. Compared with the original representation, the human pose data represented by the Lie algebra is more conducive to a calculation, and we use the standard bone length to replace the original bone length, excluding the influence of bone length on classification.

Then, we implement and train networks by Pytorch [35]. The result is shown in Table 1. We adopt the network architecture in [14], which is combined with three LSTM networks and seven CNN networks. We made a small change in the front part of the network structure. For the input to the LSTM network, our structure contains the transformed Lie algebra. At the same time, we also adopted

the method of calculating a TPF for the input of the CNN network. We trained the weights of the network on the software and obtained an accuracy rate close to Ref. [14].

Table 1. Experimental results on H3.6M and NTU RGB+D datasets.

Dataset	Method	Cross Subject	Cross View	Accuracy
H3.6M	All-Mul-Score fusion (LSTM+CNN) (Software Implementation)	81.78%	88.97%	86.31%
	All-Mul-Score fusion (LSTM+CNN) (Memristor Simulation)	80.67%	88.44%	85.98%
NTU RGB+D	All-Mul-Score fusion (LSTM+CNN) (Software Implementation)	*82.78%*	*91.13%*	*86.53%*
	All-Mul-Score fusion (LSTM+CNN) (Memristor Simulation)	*79.80%*	*87.97%*	*83.31%*

In this section of our study, we employed a simulated memristor architecture using MemTorch [36], which is a simulation platform for memristive deep learning systems that seamlessly integrates with the PyTorch machine learning (ML) library. MemTorch facilitates the emulation of memristor crossbars and allows for a direct interaction with PyTorch. This integration enables the straightforward mapping of network structures implemented in PyTorch, including LSTM and CNN networks, onto the crossbar architecture. Furthermore, we leveraged MemTorch's capability to map both the network structure and the weights for a simulated inference, opting to utilize perfect-state memristor structures despite MemTorch's support for modeling imperfect memristor properties.

Based on existing literature [21,24,37,38], it is crucial to highlight that employing actual memristor structures could not only significantly reduce the power consumption, but also accelerate the inference speeds compared to the simulated memristor structures utilized in our study. However, within the scope of this research, we have chosen not to empirically demonstrate this potential for an enhanced efficiency and speed. This decision was made to maintain our focus on the simulation aspects of memristor-based systems, given the constraints of our experimental setup.

3. Analysis of proposed method

3.1. Challenges and limitations

The proposed method introduces several challenges and limitations that need careful consideration. First, while it is beneficial for computational efficiency, the conversion of human pose data into a Lie algebra representation adds a layer of complexity in the data interpretation that may obscure intuitive insights. This complexity is compounded by the use of standard bone lengths, which, although eliminating individual anatomical variations, may limit the model's ability to generalize across diverse body shapes and sizes.

Moreover, the modifications made to the network architecture to accommodate the Lie algebra

inputs and the computation of temporal-domain-features, though innovative, may not be universally optimal. The performance of these architectural changes could significantly vary, necessitating extensive empirical validation. Additionally, the reliance on perfect-state memristor structures in MemTorch simulations may not accurately reflect the imperfections of real-world memristor behavior, potentially leading to an overestimation of the model's performance in practical applications. Finally, the computational resource demands for integrating LSTM and CNN networks with MemTorch are substantial, which could limit the scalability of the proposed method, especially for large-scale or real-time processing applications.

3.2. Suggestions for future research

Addressing the aforementioned challenges necessitates a multifaceted approach in future research endeavors. It is imperative to explore alternative mathematical representations of human pose data that balance the computational efficiency with the ability to intuitively interpret and generalize across different body types. Developing adaptive network architectures that can dynamically adjust to the specific characteristics of the input data could potentially enhance the performance across a broader range of scenarios.

Incorporating more realistic models of memristor behavior into simulations is crucial for accurately anticipating the challenges associated with deploying these systems in real-world settings. Furthermore, optimizing the computational efficiency of the proposed method through either algorithmic improvements or the adoption of more efficient hardware is essential for enhancing the scalability and enabling real-time processing capabilities. Finally, conducting extensive validation studies across a variety of datasets is vital to thoroughly evaluate the generalizability and robustness of the proposed method, thus identifying areas that require further refinement.

4. Conclusions

This study explored the application of memristor-based circuits to simulate neural networks in the context of human action recognition using skeletal data. By leveraging Lie algebra and standardized bone length data for an efficient representation of human skeletons, we demonstrated the feasibility of using memristor technology to approximate the functionality of multi-layer LSTM recurrent neural networks combined with CNNs. Our work contributes to the field by showcasing a novel use of memristor circuits for network inference, which offers a promising avenue for reducing energy consumption and accelerating inference times in deep learning models.

A pivotal aspect of our research focuses on the construction of networks using memristor circuits, which are capable of achieving performance metrics closely approximating those of software-simulated networks. Although our memristor network implementation remained within the realm of the simulation, the inherent efficiency and low power consumption of memristor structures have been well-documented. This approach not only addresses critical challenges in deploying deep learning models for real-time applications, but also highlights the potential of the memristor technology as a sustainable and efficient computing alternative to traditional, power-intensive computational methods.

Furthermore, we illustrated that it is possible to maintain a balance between the computational efficiency and the model accuracy, which is often a significant challenge in optimizing deep learning models. The ability to achieve near-original performance metrics with memristor-based simulations

underscores the potential of our method for a broad application in various sectors, including healthcare, autonomous driving, surveillance, and sports analytics.

In conclusion, our research highlights the viability of memristor-based deep learning systems for human action recognition, marking a step towards the practical implementation of energy-efficient and fast neural network simulations. The implications of our work are far-reaching, suggesting a future where memristor technologies can play crucial roles in enabling real-time, energy-efficient, and accurate computational tasks across diverse applications.

Author contributions

Zhencheng Fan: Conceptualized the study, developed the methodology, and was involved in project administration; Zheng Yan: Contributed to the methodology, handled data curation, and performed the formal analysis; Yuting Cao: Involved in software validation, conducted the formal analysis, and assisted in writing the original draft; Yin Yang: Contributed to the investigation process, curated data, and reviewed and edited the manuscript; Shiping Wen: Supervised the project, acquired funding, and was responsible for the final review and editing of the manuscript. All authors have read and approved the final version of the manuscript for publication.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work is supported by NPRP grant: NPRP 9-466-1-103 from Qatar National Research Fund. The statements made herein are solely the responsibility of the authors.

Conflict of interest

Shiping Wen is an guest editor for [Dynamic analysis and scientific application of time-delay neural networks] and was not involved in the editorial review or the decision to publish this article. All authors declare that there are no competing interests.

References

1. J. Rafferty, C. D. Nugent, J. Liu, L. Chen, From activity recognition to intention recognition for assisted living within smart homes, *IEEE Trans. Human Machine Syst.*, **47** (2017), 368–379. https://doi.org/10.1109/THMS.2016.2641388
2. Y. Sun, Z. Zhang, I Kakkos, G. K. Matsopoulos, J. J. Yuan, J. Suckling, Inferring the individual psychopathologic deficits with structural connectivity in a longitudinal cohort of Schizophrenia, *IEEE J. Biomed. Health Informa.*, **26** (2022), 2536–2546. https://doi.org/10.1109/JBHI.2021.3139701

3. Z. Guo, L. Zhao, J. Yuan, H. Yu, MSANet: Multiscale aggregation network integrating spatial and channel information for Lung nodule detection, *IEEE J. Biomed. Health Inform.*, **26** (2022), 2547–2558. https://doi.org/10.1109/JBHI.2021.3131671
4. J. W. Li, S. Barma, P. Un Mak, F. Chen, C. Li, M. T. Li, et al., Single-channel selection for EEG-based emotion recognition using brain rhythm sequencing, *IEEE J. Biomed. Health Inform.*, **26** (2022), 2493–2503. https://doi.org/10.1109/JBHI.2022.3148109
5. C. Finn, I. Goodfellow, S. Levine, Unsupervised learning for physical interaction through video prediction, *arXiv:1605.07157*, 2016. https://doi.org/10.48550/arXiv.1605.07157
6. L. Liu, L. Cheng, Y. Liu, Y. Jia, D. S. Rosenblum, Recognizing complex activities by a probabilistic interval-based model, In: *Proceedings of the thirtieth AAAI conference on artificial intelligence (AAAI'16)*, AAAI Press, 2016, 1266–1272. https://doi.org/10.5555/3015812.3015999
7. Z. Cao, T. Simon, S. E. Wei, Y. Sheikh, Realtime multi-person 2D pose estimation using part affinity fields, *arXiv:1611.08050*, 2016. https://doi.org/10.48550/arXiv.1611.08050
8. R. Vemulapalli, F. Arrate, R. Chellappa, Human action recognition by representing 3D skeletons as points in a Lie group, In: *2014 IEEE Conference on computer vision and pattern recognition*, 2014, 588–595. https://doi.org/10.1109/CVPR.2014.82
9. K. Fragkiadaki, S. Levine, P. Felsen, J. Malik, Recurrent network models for human dynamics, In: *IEEE International conference on computer vision (ICCV)*, 2015, 4346–4354. https://doi.org/10.1109/ICCV.2015.494
10. A. Jain, A. R. Zamir, S. Savarese, A. Saxena, Structural-RNN: Deep learning on spatio-temporal graphs, In: *2016 IEEE Conference on computer vision and pattern recognition (CVPR)*, 2016, 5308–5317. https://doi.org/10.1109/CVPR.2016.573
11. J. Redmon, A. Farhadi, YOLOv3: An incremental improvement, *arXiv:1804.02767*, 2018. https://doi.org/10.48550/arXiv.1804.02767
12. K. Smagulova, A. P. James, A survey on LSTM memristive neural network architectures and applications, *Eur. Phys. J. Spec. Top.*, **228** (2019), 2313–2324. https://doi.org/10.1140/epjst/e2019-900046-x
13. J. Hu, Z. Fan, J. Liao, L. Liu, Predicting long-term skeletal motions by a spatio-temporal hierarchical recurrent network, *arXiv:1911.02404*, 2019. https://doi.org/10.48550/arXiv.1911.02404
14. C. Li, P. Wang, S. Wang, Y. Hou, W. Li, Skeleton-based action recognition using LSTM and CNN, In: *2017 IEEE International conference on multimedia & expo workshops (ICMEW)*, 2017, 585–590. https://doi.org/10.1109/ICMEW.2017.8026287
15. Q. Huang, L. Jia, G. Ren, X. Wang, C. Liu, Extraction of vascular wall in carotid ultrasound via a novel boundary-delineation network, *Eng. Appl. Artif. Intell.*, **121** (2023), 106069. https://doi.org/10.1016/j.engappai.2023.106069
16. J. Liu, Y. Wang, Y. Liu, S. Xiang, C. Pan, 3D PostureNet: A unified framework for skeleton-based posture recognition, *Pattern Recognition Lett.*, **140** (2020), 143–149. https://doi.org/10.1016/j.patrec.2020.09.029

17. P. Wang, J. Wen, C. Si, Y. Qian, L. Wang, Contrast-reconstruction representation learning for self-supervised skeleton-based action recognition, *IEEE Trans. Image Process.*, **31** (2022), 6224–6238. https://doi.org/10.1109/TIP.2022.3207577

18. A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet classification with deep convolutional neural networks, *Commun. ACM*, **60** (2017), 84–90. https://doi.org/10.1145/3065386

19. T. M. Taha, R. Hasan, C. Yakopcic, M. R. McLean, Exploring the design space of specialized multicore neural processors, In: *2013 International joint conference on neural networks (IJCNN)*, 2013, 1–8. https://doi.org/10.1109/IJCNN.2013.6707074

20. L. Chua, Memristor-The missing circuit element, *IEEE Trans. Circuit Theory*, **18** (1971), 507–519. https://doi.org/10.1109/TCT.1971.1083337

21. S. Wen, R. Hu, Y. Yang, T. Huang, Z. Zeng, Y. D. Song, Memristor-based echo state network with online least mean square, *IEEE Trans. Syst. Man Cybernet.*, **49** (2019), 1787–1796. https://doi.org/10.1109/TSMC.2018.2825021

22. S. H. Jo, K. H. Kim, W. Lu, High-density crossbar arrays based on a Si memristive system, *Nano Lett.*, **9** (2009), 870–874. https://doi.org/10.1021/nl8037689

23. R. Hasan, T. M. Taha, C. Yakopcic, On-chip training of memristor crossbar based multi-layer neural networks, *Microelectronics J.*, **66** (2017), 31–40. https://doi.org/10.1016/j.mejo.2017.05.005

24. S. Wen, H. Wei, Y. Yang, Z. Guo, Z. Zeng, T. Huang, et al., Memristive LSTM network for sentiment analysis, *IEEE Trans. Syst. Man Cybernet.*, **51** (2019), 1794–1804. https://doi.org/10.1109/TSMC.2019.2906098

25. X. Liu, Z. Zeng, D. C. Wunsch, Memristor-based LSTM network with in situ training and its applications, *Neural Netw.* **131** (2020), 300–311. https://doi.org/10.1016/j.neunet.2020.07.035

26. C. Yakopcic, M. Z. Alom, T. M. Taha, Memristor crossbar deep network implementation based on a convolutional neural network, In: *2016 International joint conferenceon neural networks (IJCNN)*, IEEE, 2016, 963–970. https://doi.org/10.1109/IJCNN.2016.7727302

27. C. Yakopcic, M. Z. Alom, T. M. Taha, Extremely parallel memristor crossbar architecture for convolutional neural network implementation, In: *2017 International joint conference on neural networks (IJCNN)*, IEEE, 2017, 1696–1703. https://doi.org/10.1109/IJCNN.2017.7966055

28. S. Wen, J. Chen, Y. Wu, Z. Yan, Y. Cao, Y. Yang, CKFO: Convolution kemel first operated algorithm with applications in memristor-based convolutional neural network, *IEEE Trans. Comput. Design Integr. Circuits Syst.*, **40** (2020), 1640–1647. https://doi.org/10.1109/TCAD.2020.3019993

29. P. Yao, H. Wu, B. Gao, J. Tang, Q. Zhang, W. Zhang, et al., Fully hardware-implemented memristor convolutional neural network, *Nature*, **577** (2020), 641–646. https://doi.org/10.1038/s41586-020-1942-4

30. C. Ionescu, D. Papava, V. Olaru, C. Sminchisescu, Human3.6M: Large scale datasets and predictive methods for 3d human sensing in natural environments, *IEEE Trans. Pattern Anal. Machine Intell.*, **36** (2013), 1325–1339. https://doi.org/10.1109/TPAMI.2013.248

31. A. Shahroudy, J. Liu, T. T. Ng, G. Wang, NTU RGB+D: A large scale dataset for 3D human activity analysis, In: *2016 IEEE Conference on computer vision and pattern recognition (CVPR)*, IEEE, 2016, 1010–1019. https://doi.org/10.1109/CVPR.2016.115

32. Y. Du, W. Wang, L. Wang, Hierarchical recurrent neural network for skeleton based action recognition, In: *2015 IEEE Conference on computer vision and pattern recognition (CVPR)*, IEEE, 2015, 1110–1118. https://doi.org/10.1109/CVPR.2015.7298714

33. C. Li, Y. Hou, P. Wang, W. Li, Joint distance maps based action recognition with convolutional neural networks, *IEEE Signal Process. Lett.*, **24** (2017), 624–628. https://doi.org/10.1109/LSP.2017.2678539

34. P. Wang, W. Li, C. Li, Y. Hou, Action recognition based on joint trajectory maps with convolutional neurall networks, *Knowledge Based Syst.*, **158** (2018), 43–53. https://doi.org/10.1016/j.knosys.2018.05.029

35. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, et al., PyTorch: An imperative style, high-performance deep learning library, In: *Proceedings of the 33rd international conference on neural information processing systems*, 2019, 8026–8037.

36. C. Lammie, W. Xiang, B. Linares-Barranco, M. R. Azghadi, MemTorch: An open-source simulation framework for memristive deep learning systems, *Neurocomputing*, **485** (2022), 124–133. https://doi.org/10.1016/j.neucom.2022.02.043

37. Hadiyawarman, F. Budiman, D. G. O. Hernowo, R. R. Pandey, H. Tanaka, Recent progress on fabrication of memristor and transistor-based neuromorphic devices for high signal processing speed with low power consumption, *Jpn. J. Appl. Phys.*, **57** (2018), 03EA06. https://doi.org/10.7567/JJAP.57.03EA06

38. S. S. Sarwar, S. A. N. Saqueb, F. Quaiyum, A. B. M. H. U. Rashid, Memristor-based nonvolatile random access memory: Hybrid architecture for low power compact memory design, *IEEE Access*, **1** (2013), 29–34. https://doi.org/10.1109/ACCESS.2013.2259891

AIMS Press