Analysis of Input Data Configurations in CNN-based Human Action Recognition for Assembly Task*

Cosimo Patruno[†1], Grazia Cicirelli[1], Laura Romeo[1], Tiziana D'Orazio[1]

***Abstract*— Human Action Recognition (HAR) plays a vital role in manufacturing assembly tasks, addressing key areas such as worker safety, operational support, production optimization, employee training, and facilitating human-robot collaboration. This paper introduces a skeleton-based action recognition approach based on a CNN deep neural network architecture. Joint-to-joint distances are used to represent human movements during assembly tasks, enabling the model to capture intricate motion patterns. The primary focus of this work is on structuring the input data in various ways to analyze how these variations influence the network performance. Studying the spatial configurations of input data for human action recognition in an assembly task is an insightful and challenging research topic. In assembly tasks, the high similarity between actions and the operator-specific execution variations make distinguishing actions more complex. This work investigates how the arrangement of input data impacts model accuracy. In particular, two input data configurations are analyzed: one-channel and *multi-channel* types. The assembly actions are classified using a CNN-based architecture. So, the different data configurations directly influence the type of CNN applied, which can be 2D or 3D. The proposed approach is evaluated on the publicly available HA4M dataset. The obtained results showed that the proposed data structure greatly influences the model performance measurement.**

I. INTRODUCTION

In recent years, Human Action Recognition (HAR) has gained increasing interest in the literature due to its crucial role in several real-world applications, such as visual surveillance, human-robot interaction, healthcare, and entertainment [1], [2], [3]. In the field of smart manufacturing, HAR plays a significant role as it reflects the ongoing transformation towards Industry 4.0, and it serves several challenging purposes, such as worker safety and support, production improvement, employee training, analysis and optimization of human activities, and so on [4], [5], [6]. Specifically, assembly tasks represent a fundamental aspect of many manufacturing processes, so developing advanced HAR systems is very useful for pursuing efficient, adaptive, and human-centered production systems. HAR enables robots to perceive, interpret, and respond to human behaviors, and in manufacturing, HAR is essential for monitoring worker activity, ensuring safety, and enabling adaptive robotic responses.

HAR in this context is challenging as assembly tasks often involve repetitive and precise operations, similar or task-specific actions, manipulation of tools and parts, variations in individual workers' abilities, and specific environmental factors such as lighting and occlusions. In the last decades, the literature has witnessed a significant increase in research focused on various approaches to HAR in assembly tasks, spanning from vision-based methods to sensor-based techniques or hybrid approaches, each offering unique advantages and challenges.

Vision-based methods for HAR in assembly tasks primarily rely on RGB, RGB-D, RGB-T, and other video data captured by cameras and sensors to interpret and classify human actions. Moreover, the recent spread of deep learning techniques for HAR in assembly tasks has revolutionized the field due to their ability to automatically extract and classify features from large amounts of visual data [7], [8]. Deep Neural Networks have become the dominant machine learning approach that enables highly accurate recognition of complex actions in dynamic environments [9], [10].

In the literature, HAR in assembly tasks mainly relies on image data and deep learning models [11], [12], [13], achieving high recognition accuracy even in complex and dynamic scenarios, such as industrial settings. However, these approaches can face challenges related to occlusions, varying lighting conditions, scale variations, camera movements and placement, differences in human body proportions, and privacy concerns [14].

In contrast, skeleton-based approaches, which focus on skeletal representations rather than raw video footage, provide a more abstract and privacy-friendly approach to monitor human actions in industrial assembly tasks. These methods involve tracking key human body joints over time to recognize actions, using data from depth cameras, 3D sensors, or computer vision algorithms. Skeleton abstraction helps preserve individual privacy and anonymity. Moreover, computational complexity is reduced as data dimensionality significantly decreases, focusing only on essential motion features. This makes these methods efficient and suitable for real-time applications, such as assembly tasks in smart manufacturing.

Skeleton joints of the human body are utilized in [15], [16], together with deep neural network architectures, to recognize assembly actions. The main goal is to optimally capture spatial and temporal relationships within motion data. However, much of the research focuses on developing

*This research has been partly funded by PNRR - M4C2 - Investimento 1.3, Partenariato Esteso PE00000013 - "FAIR - Future Artificial Intelligence Research" - Spoke 8 "Pervasive AI", funded by the European Commission under the NextGeneration EU program.

[1]C. Patruno, G. Cicirelli, L. Romeo and T. D'Orazio are with the Institute of Intelligent Industrial Systems and Technologies for Advanced Manufacturing (STIIMA) of the National Research Council (CNR), Bari, Italy.

[†]Corresponding Author: cosimo.patruno@cnr.it

or improving more complex machine learning or deep learning models rather than exploring adjustments to input data that could achieve comparable or even better performance.

This paper proposes a skeleton-based HAR method with a particular focus on how different input data structuring affects model accuracy [17], [18]. This is a challenging research topic that has been poorly investigated in the context of HAR in assembly tasks. Literature works usually encode the spatial features of skeleton poses into image representations that integrate both spatial and temporal information [19], [20], [21]. Our aim is to explore the capability of a simple Convolutional Neural Network (CNN) to capture spatial and temporal dependencies in data without relying on those models specifically designed to process sequential data, like recurrent neural networks. The use of CNN networks is advantageous for their computational efficiency and simplified architectures, making them less resource-intensive than more complex models like Recurrent Neural Networks. Our tests are carried out on the HA4M assembly dataset [22], which includes fine-grained and similar actions and provides multimodal data, including skeleton joint coordinates. We use joint-to-joint distances to represent spatial motion during the assembly task, whereas a sliding window approach is applied to extract time window series, retaining the essential motion and temporal information. A CNN-based deep network architecture is trained from scratch to recognize the assembly actions by analyzing different arrangements of input features. Two input data configurations are analyzed: *one-channel* and *multi-channel* types. These configurations directly influence the CNN architecture: 2D or 3D. Therefore, the presented study contributes to a twofold objective: the impact of input data structuring and how to process these data, either using a 2D or a 3D model. The approach demonstrates high recognition rates, proving that the choice of structuring input data in different ways can significantly influence model performance. Our study suggests that models built on *multi-channel* configuration are more effective for recognizing actions in an assembly task.

The remainder of the paper is structured as follows. Section II describes the proposed method, giving details on the input data and the network architecture. Then, experimental results are provided and discussed in Section III. Section IV concludes the paper, focusing on future research directions.

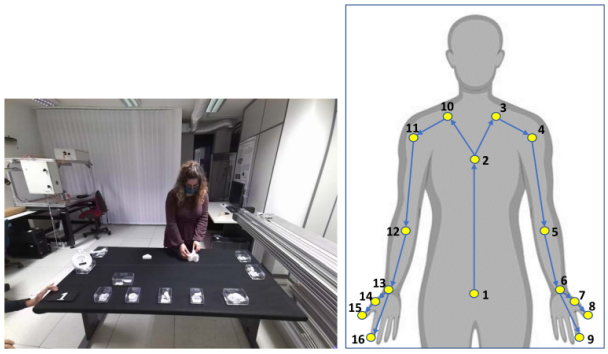
II. METHOD

This section gives in-depth details about the dataset used for validating our proposal, the computed features to feed the deep learning models, the different ways in which the input data can be arranged, and the CNN-based architectures.

A. Dataset

This study has been tested using the HA4M dataset [22]. This is a collection of multi-modal data related to actions performed by various subjects while assembling an industrial object. The data was recorded in a laboratory setting using a Microsoft Azure Kinect. The HA4M dataset provides various data types, including RGB frames, IR frames, depth maps, RGB-to-depth-aligned frames, point clouds, and Skeleton data.

In this work, we use the skeleton data. Skeleton data represents a person's body as a set of key points called joints without the need to capture identifiable visual characteristics. As a result, skeleton-based HAR preserves individual privacy and anonymity while accurately recognizing human actions.



(a) (b)

Fig. 1. (a) Sample frame of HA4M dataset: the operator stands behind a desk, where the components to be assembled are spread over. (b) Joints of the upper body.

This study has been tested using the HA4M dataset [22]. This is a collection of multi-modal data related to actions performed by various subjects while assembling an industrial object. The data was recorded in a laboratory setting using a Microsoft Azure Kinect. The HA4M dataset provides various data types, including RGB frames, IR frames, depth maps, RGB-to-depth-aligned frames, point clouds, and Skeleton data.

In this work, we use the skeleton data. Skeleton data represents a person's body as a set of key points called joints without the need to capture identifiable visual characteristics. As a result, skeleton-based HAR preserves individual privacy and anonymity while accurately recognizing human actions.

B. Features

In this study, the distance between pairs of skeletal joints has been calculated to identify key features for capturing the subjects' posture during assembly task execution. These distances have been computed for every frame in the video, allowing us to maintain precise joint correlations. The total number of distances for each frame is $N = \binom{M}{2}$. To account for variations in the subjects' body sizes and differences in camera placement and orientation, the distance features have been normalized to make them invariant to these factors. Given two different skeletal joints j and k, with $j \neq k$, in frame F^i, identified by 3D points $J_j^i = (X_j^i, Y_j^i, Z_j^i)$ and $J_k^i = (X_k^i, Y_k^i, Z_k^i)$ respectively, the distance feature $d_{j,k}^i$ can be calculated as follows:

$$d_{j,k}^i = \frac{\|J_j^i - J_k^i\|_2}{d_{Norm}^i} \qquad \forall i \tag{1}$$

where

$$d_{Norm}^i = d_{4,11}^i + d_{1,2}^i$$

represents the normalization factor, which is defined as the sum of two distances: 1) the distance $d^i_{4,11} = \|J^i_4 - J^i_{11}\|_2$ between the joints of the right and left shoulders (joints no. 4 and no. 11 in Figure 1(b)) and 2) the distance $d^i_{1,2} = \|J^i_1 - J^i_2\|$ between the joints of the pelvis and the spine chest (joints no. 1 and no. 2 in Figure 1(b)).

The normalized distance features $d^i_{j,k}$ calculated over the frame F^i can be structured in different matrix forms, as discussed in the following section.

C. Input Data Structuring

The assembly task within the HA4M dataset comprises a sequence of $N_A = 12$ actions that 41 operators perform at their discretion and preferred, comfortable pace. Each of these actions exhibits varying levels of complexity, resulting in different durations, thus leading to a notable time variance among actions. These factors were taken into account when defining the input data to the neural network architecture.

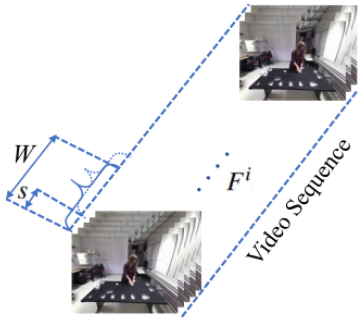
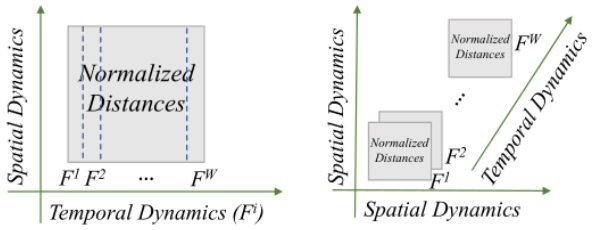


Fig. 2. Sliding window approach to extract the spatio-temporal skeletal data. The sliding window has a length of W frames and moves along the timeline of the video sequence with a stride size s.

In particular, a sliding-window approach has been adopted to extract spatio-temporal skeletal data pieces. The sliding window, having a length of W frames and a stride size of s frames, moves along the timeline of the video sequence as shown in Figure 2. Determining the appropriate W for the sliding window size is crucial, striking a balance with the length of the actions. A small window may not capture sufficient information to characterize the actions effectively. On the other hand, a large window might span a considerable time interval, encompassing data from multiple actions. Finally, the choice of W for the sliding window size has been fixed to 15, considering the minimum length of the actions and the frame rate of the Azure Kinect, which is 30 fps. Therefore, the sliding window covers a time duration of half a second.

Once extracted the W frames from the video sequence and obtained the relative skeletons, the normalized distances $d^i_{j,k}$, with $i = 1...W$, $j, k = 1...M$, and $j \neq k$, can be used to construct the input data tensor, which will be fed into the deep neural network for action recognition.

The input tensor to a CNN can have two principal structures: 1) *one-channel* structure, so the tensor is a matrix with spatial information on the matrix rows and temporal information on the matrix columns (depth=1); 2) *multi-channel* structure, where the tensor has spatial information over rows and columns, and temporal information on a number of channels (depth$>$ 1). Figure 3 graphically shows these concepts.



(a) (b)

Fig. 3. Structures of input data: (a) *One-channel* case (depth = 1). (b) *Multi-channel* case (depth = W).

In the *one-channel* case, the input tensor is a 2D matrix $N \times W$ where each column D^i refers to each frame F^i and contains the N normalized distances as follows:

$$D^i = \begin{bmatrix} d^i_{1,2} \\ \vdots \\ d^i_{M-1,M} \end{bmatrix} \quad \forall i = 1...W$$

In the *multi-channel* case, instead, the input tensor for the CNN has dimension $H \times K \times W$, where W represents the depth and is determined by the size of the sliding window, which captures the temporal information; H and K define the spatial dimension and vary depending on the arrangements of the N distance features. Table I lists the different tensor dimensions considered in this study. In particular, in both cases, *one-channel* and *multi-channel*, we have defined the full and reduced dimensions for the input tensor. The full dimension means that all the available normalized distances have been used to build the input tensor. Notice that since the number of selected joints is $M = 16$, the possible joint-to-joint distances are $N = \binom{16}{2} = 120$. The reduced dimension involves a smaller subset of distance features that we have selected by task-specific semantic selection. In the assembly task, in fact, human actions are mainly repetitive movements of the arms. Therefore, we have selected N_r joint-to-joint distances that refer to the distances within the arms and those between each arm and the torso joints, which capture interactions with the body.

TABLE I

DIMENSION OF THE INPUT TENSOR IN THE DIFFERENT CONFIGURATIONS

Case		Tensor Dimension
one-channel case	Full	120×15 $(N \times W)$
	Reduced	70×15 $(N_r \times W)$
multi-channel case	Full	$16 \times 16 \times 15$ $(M \times M \times W)$
	Full	$12 \times 10 \times 15$ $(H \times K \times W)$
	Reduced	$7 \times 10 \times 15$ $(H_r \times K_r \times W)$

In the *multi-channel* case, the spatial information derived from each frame can be configured in various ways. We have considered two configurations for the full dimension, $(M \times M \times W)$ and $(H \times K \times W)$, and one for the reduced dimension, $(H_r \times K_r \times W)$. Considering every single frame F^i within the W frames of the sliding windows, all the joint-to-joint distances can be structured in a matrix $(M \times M)$, where M is the number of joints and each element $d^i_{j,k}$ of the matrix is the distance between the joint i and the joint j. The configuration $(H \times K \times W)$ includes the 120 distances structured in 12 rows and 10 columns. Finally, for the reduced dimension, we have considered the same subset of distances selected in the *one-channel* case but arranged in the (7×10) configuration.

D. Neural Network Architecture

This section describes the CNN-based neural network architectures used to build the HAR model. The architectures are inspired by the AlexNet CNN model [23], a pioneering deep learning architecture widely used for classification problems. Figure 4 shows the proposed network architectures, which consist of six layers: the first four are convolutional layers, of which the second and fourth are followed by a max-pooling layer. Figure 4 also lists the details of the network configuration in terms of kernel size (Kr), stride size (St), padding size (Pd), and the number of filters (Ft).

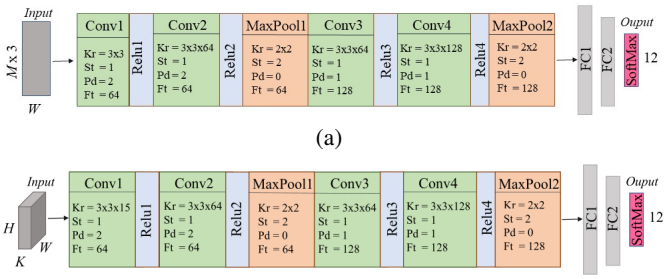


Fig. 4. Architectures based on Convolutional neural networks (CNNs). (a) *One-channel* case: the input tensor has $N \times W$ dimension. (b) *Multi-channel* case: the input tensor has$(H \times K \times W)$ dimension.

Each convolutional layer generates a feature map using the ReLU activation function. The first and second convolutional layers have 64 channels (feature maps), while the subsequent two convolutional layers have 128 channels. Max-pooling layers are applied to reduce the spatial dimensions of the feature maps by a factor of two. The final feature map is then processed through two fully connected layers, transforming it into a $1 \times N_A$ score vector, which is subsequently converted into a probabilistic action class prediction using the softmax activation function.

The network architectures shown in Figure 4 differ only in the first layer, which determines their 2D and 3D variants. As can be noticed in Figure 4(a), the convolutional filters at the first layer are 2D matrices that can slide vertically and horizontally across the 2D input matrix to extract local features. In contrast, Figure 4(b) shows the 3D CNN, where convolutional filters have three dimensions to process volumetric input data, represented in this case by a stack of 2D matrices over time. Thus, the key distinction between the 2D and 3D architectures applied in this work lies in the dimensionality of the convolutional filters at the first layer of the network and the structure of the input data, which are strictly interconnected. The rest of the layers are the same for both network architectures.

III. EXPERIMENTS

This section describes the experimental results of action recognition using the *one-channel* and *multi-channel* input data configurations and thus the 2D and 3D CNN architectures as defined in section II-D. First, major details about the considered input data are given, and then the results are shown and discussed.

A. Data and parameter definition

The proposed approach has been evaluated using a subset of the HA4M dataset [22]. This subset consists of 49200 tensors, opportunely selected to ensure a balanced distribution of samples across different actions and operators performing the task. The dataset has been split considering the 60% for training (29520 items), the 10% for validation (4920 items), and the remaining 30% (14760 items) for testing. Notice that the same number of samples has been generated for each configuration listed in Table I.

The network architectures have been trained on an NVIDIA GeForce RTX 3070 GPU with 8 GB of dedicated memory. Training has been performed using stochastic gradient descent with momentum (SGDM) as the optimization solver. The initial learning rate was set to 0.5, with a momentum value of 0.9. A batch size of 256 was used, and the training set was shuffled at every epoch. All experiments were conducted within the MATLAB framework.

TABLE II
METRICS DEFINITION. TP, TN, FP, AND FN STAND FOR TRUE POSITIVES, TRUE NEGATIVES, FALSE POSITIVES, AND FALSE NEGATIVES, RESPECTIVELY.

Accuracy	**Precision**
$\frac{TP+TN}{TP+FP+TN+FN}$	$\frac{TP}{TP+FP}$
Recall	**F-score**
$\frac{TP}{TP+FN}$	$\frac{TP}{TP+\frac{1}{2}(FP+FN)}$

B. Results

This section presents the action recognition results obtained by considering the different input tensor configurations and applying the convolutional-based deep architectures described in section II-D. Training was carried out for a maximum of 500 epochs; however, early stopping was applied by monitoring accuracy and loss on the training and

TABLE III

QUANTITATIVE COMPARISON OF THE DIFFERENT CASES ANALYZED IN THIS WORK. THE VALUE IN BOLD DENOTES THE BEST ACCURACY OF ACTION RECOGNITION. GREY ROWS INDICATE THE BEST INPUT CONFIGURATION FOR THE *one-channel* AND THE *multi-channel* CASE.

Case	Model	TP	FP - FN	TN	Precision [%]	Recall [%]	F-score [%]	Accuracy [%]
one-channel								
120×15	CNN	11820	2940	159420	80.11	80.08	80.09	80.08
70×15	CNN	11580	3180	159180	78.44	78.46	78.45	78.46
120×15	LSTM	10649	4111	158249	72.16	72.15	72.15	72.15
70×15	LSTM	9966	4794	157566	67.62	67.52	67.57	67.52
multi-channel								
$16 \times 16 \times 15$	CNN	12602	2158	160202	85.39	85.38	85.38	85.38
$12 \times 10 \times 15$	CNN	12630	2130	160230	85.62	85.57	85.59	**85.57**
$7 \times 10 \times 15$	CNN	12529	2231	160129	84.92	84.88	84.90	84.88
$12 \times 10 \times 15$	LSTM	10893	3867	158493	73.85	73.80	73.82	73.80
$7 \times 10 \times 15$	LSTM	9965	4795	157565	67.53	67.51	67.52	67.51

validation sets, halting the process when no significant variations in the loss were observed. In general, training accuracy exceeded 98%. Moreover, the evolution of the accuracy of the training set and validation set were observed to avoid overfitting problems. After training, model performance was evaluated on the test set that is entirely disjoint from the training and validation sets. In particular, confusion matrices were analyzed for all cases, and key performance metrics were extracted. These metrics, defined in Table II, include Accuracy, Precision, Recall, and F-score. Accuracy measures the proportion of correctly classified samples (TP and TN) out of all test samples. Precision quantifies the proportion of correctly predicted positive samples (TP) among all predicted positive samples (TP and FP). Recall represents the proportion of correctly predicted positive samples (TP) out of all actual positive samples (TP and FN). Finally, the F-score is the harmonic mean of Precision and Recall, providing a balanced measure of model performance.

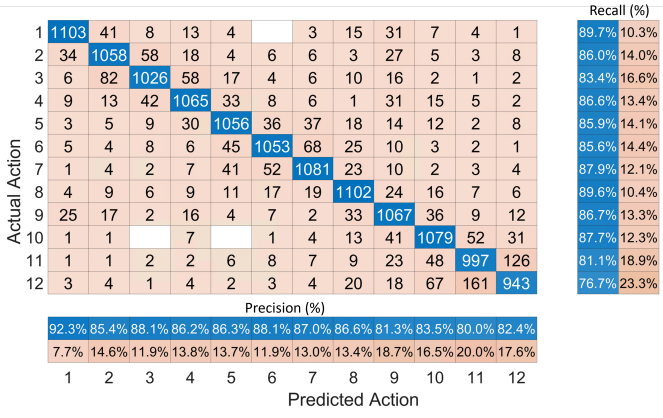


Fig. 5. Confusion Matrix relative to the *multi-channel* case with full tensor dimension $12 \times 10 \times 15$.

Table III lists the aforementioned metrics evaluated in each case: (*one-channel* and *multi-channel*). As can be seen in the table, the best performance is generally achieved with the *multi-channel* configuration, regardless of whether the input tensor has full or reduced dimensions. This indicates that the information included in the temporal channels of the input tensor, along the third tensor component, is important for a more accurate recognition of actions with respect to the *one-channel* case. Finally, additional experiments were carried out by applying an LSTM (Long Short-Term Memory) neural network to further investigate the study. The used LSTM-based model is composed of an LSTM layer having 128 hidden units, a fully-connected layer having an output size of 12 elements, and a softmax layer. However, recognition performance deteriorated in both cases by more than 8%. Despite being specifically designed to capture temporal dependencies in sequential data, LSTMs struggled in this particular context. They are unable to recognize actions effectively when limited to observation windows of just half a second and fail to capture meaningful spatial relationships among frames.

For completeness, Figure 5 shows the confusion matrix relative to the best performance case, i.e. the *multi-channel* case with full tensor dimension $12 \times 10 \times 15$.

C. Discussion

The presented study and the experimental results demonstrate that different input tensor configurations in a CNN-based deep neural network can have a considerable impact on action recognition performance. Clearly, in the particular context of assembly action recognition, the *multi-channel* case and, consequently, the 3D CNN models provide better results. The accuracy rate, in fact, ranges from 84.88% for the reduced input tensor dimension to 85.57% for the full tensor dimension. The accuracy in the *one-channel* case, instead, reaches 80.08% and 78.46% for the full and reduced input tensor dimension, respectively. However, the use of reduced tensor dimensions is notable, especially in those applications where lower computational costs are required.

For what concerns the obtained results, some key considerations can be outlined. In the *multi-channel* case, the meaningful features are captured along the temporal evo-

lution of actions that can be observed along the channels of the convolutional layers. At the same time, the spatial relationships are well captured within the 2D matrices built from each frame. In contrast, in the *one-channel* case, spatial information from each frame is configured into a 1D vector, which may limit the network's ability to capture spatial dependencies. A crucial factor in CNN-based action recognition is the choice of kernel size, which significantly affects how the network learns spatial and temporal features. Therefore, the configurations of input tensors and the kernel size are two key strictly interconnected aspects in the design of CNN architectures for action recognition contexts where movement dynamics evolve over time.

Although CNNs can process 2D or 3D input tensors, LSTMs are restricted to 1D vectorial inputs that represent long sequences of features over time. In our particular context, LSTM performance is lower than CNN performance, with no significant difference between the *multi-channel* and *one-channel* cases. This is because for LSTM, 2D or 3D input tensors are flattened into 1D sequential inputs, losing the spatial structures that CNNs effectively manage. This observation is also supported by the various values of the metrics presented in Table III, which for LSTM show comparable results across different configurations.

IV. CONCLUSION

This paper introduces a skeleton-based action recognition approach based on a CNN architecture in an assembly task. The main issue investigated in this work is how different configurations of the input data influence CNN performance. In particular, two input data configurations are analyzed: *one-channel* and *multi-channel* types. These data configurations directly influence the type of CNN applied, which can be 2D or 3D. The proposed approach has been evaluated on the challenging HA4M dataset. The best performance was generally achieved with the *multi-channel* configuration and so with the 3D CNN, regardless of whether the input tensor had full or reduced dimensions. This indicates that the information included in the temporal channels of the input tensor is important for more accurate recognition of actions. Future research will explore additional input tensor configurations and sliding-window sizes, with a particular emphasis on optimizing the kernel size in CNN-based architectures. Furthermore, new architectures with deeper layers will be designed. Additionally, task-specific semantic feature selection will be further investigated, exploring automatic attention mechanisms to extract the most relevant features while minimizing noise from less relevant information.

REFERENCES

[1] Z. Sun, Q. Ke, H. Rahmani, M. Bennamoun, G. Wang, and J. Liu, "Human Action Recognition From Various Data Modalities: A Review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 3, pp. 3200–3225, 2023.

[2] M. Karim, S. Khalid, A. Aleryani, J. K. I. Ullah, and Z. Ali, "Human Action Recognition Systems: A Review of the Trends and State-of-the-Art," *IEEE Access*, vol. 12, pp. 36 372–36 390, 2024.

[3] C. Patruno, V. Renò, G. Cicirelli, and T. D'Orazio, "Multimodal People Re-identification using 3D Skeleton, Depth and Color Information," *IEEE Access*, vol. 12, pp. 174 689–174 704, 2024.

[4] T. Benmessabih, R. Slama, V. Havard, and D. Baudry, "Online human motion analysis in industrial context: A review," *Engineering Applications of Artificial Intelligence*, vol. 131, p. 107850, 2024.

[5] M. V. Maselli, R. Marani, G. Cicirelli, and T. D'Orazio, "Continuous Action Recognition in Manufacturing Contexts by Deep Graph Convolutional Networks," in *Lecture Notes in Networks and Systems*. Springer, 2024, vol. 825, pp. 156–173.

[6] L. Romeo, R. Marani, G. Cicirelli, and T. D'Orazio, "Multimodal data extraction and analysis for the implementation of Temporal Action Segmentation models in Manufacturing," in *CoDiT 2024, 10th International Conference on Control, Decision and Information Technologies*, La Valletta, Malta, July 2024.

[7] L. Romeo, R. Marani, A. G. Perri, and J. Gall, "Multi-modal temporal action segmentation for manufacturing scenarios," *Engineering Applications of Artificial Intelligence*, vol. 148, p. 110320, 2025.

[8] L. Romeo, C. Patruno, G. Cicirelli, and T. D'Orazio, "Multi-View Skeleton Analysis for Human Action Segmentation Tasks," in *Proc. of the 14th International Conference on Pattern Recognition Applications and Methods*, 2025, pp. 579–586.

[9] D. Q. Vu, T. P. T. Thu, N. Le, and J. C. Wang, "Deep Learning for Human Action Recognition: A Comprehensive Review," *Transactions on Signal and Information Processing*, vol. 12, no. 2, 2023.

[10] H. H. Pham, L. Khoudour, A. Crouzil, P. Zegers, and S. A. Velastin, "Video-based Human Action Recognition using Deep Learning: A Review," 2022, arXiv:2208.03775. [Online]. Available: https://arxiv.org/abs/2208.03775

[11] W. Tao, M. Al-Amin, H. Chen, M. C. Leu, Z. Yin, and R. Qin, "Real-Time Assembly Operation Recognition with Fog Computing and Transfer Learning for Human-Centered Intelligent Manufacturing," *Procedia Manufacturing*, vol. 48, pp. 926–931, 2020.

[12] A. Matin, M. R. Islam, Y. Zhu, X. Wang, H. Huo, and G. Xu, "Hybrid Deep Learning for Assembly Action Recognition in Smart Manufacturing," *International Journal of Computer Vision and Signal Processing*, vol. 14, no. 1, pp. 9–17, 2024.

[13] V. Selvaraj, M. Al-Amin, X. Yu, W. Tao, and S. Min, "Real-time action localization of manual assembly operations using deep learning and augmented inference state machines," *Journal of Manufacturing Systems*, vol. 72, pp. 504–518, 2024.

[14] I. Jegham, A. B. Khalifa, I. Alouani, and M. A. Mahjoub, "Vision-based human action recognition: An overview and real world challenges," *Forensic Science International: Digital Investigation*, vol. 32, pp. 1–17, 2020.

[15] H. Qiu and B. Hou, "Multi-grained clip focus for skeleton-based action recognition," *Pattern Recognition*, vol. 148, p. 110188, 2024.

[16] M. Al-Amin, R. Qin, M. Moniruzzaman, Z. Yin, W. Tao, and M. C. Leu, "An individualized system of skeletal data-based CNN classifiers for action recognition in manufacturing assembly," *Journal of Intelligent Manufacturing*, vol. 34, pp. 633–649, 2023.

[17] I. Toumia and A. B. Hassine, "Impact of Input Data Structure on Convolutional Neural Network Energy Prediction Model," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 10, pp. 748–757, 2022.

[18] R. Pourdarbani, S. Sabzi, R. Zohrabi, G. García-Mateos, R. Fernandez-Beltran, J. M. Molina-Martínez, and M. H. Rohban, "Comparison of 2D and 3D convolutional neural networks in hyperspectral image analysis of fruits applied to orange bruise detection," *Journal of Food Science*, vol. 88, no. 12, pp. 5149–5163, 2023.

[19] H.-H. Pham, L. Khoudour, A. Crouzil, P. Zegers, and S. A. Velastin, "Exploiting deep residual networks for human action recognition from skeletal data," *Computer Vision and Image Understanding*, vol. 170, pp. 51–66, 2018.

[20] T. Huynh-The, C. H. Hua, T. T. Ngo, and D. S. Kim, "Image representation of pose-transition feature for 3D skeleton-based action recognition," *Information Sciences*, vol. 513, pp. 112–116, 2020.

[21] C. Dai, S. Lu, C. Liu, and B. Guo, "A light-weight skeleton human action recognition model with knowledge distillation for edge intelligent surveillance applications," *Applied Soft Computing*, vol. 151, p. 111166, 2024.

[22] G. Cicirelli, R. Marani, L. Romeo, M. G. Domínguez, J. Heras, A. G. Perri, and T. D'Orazio, "The HA4M dataset: Multi-Modal Monitoring of an assembly task for Human Action recognition in Manufacturing," *Scientific Data*, vol. 9, no. 1, p. 745, 2022.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Communications of the ACM*, vol. 60, no. 6, 2017, pp. 84–90.