Skeleton-Split Framework using Spatial Temporal Graph Convolutional Networks for Action Recognition

1st Motasem S. Alsawadi
Department of Electronic and Electrical Engineering
University College London
London, UK
motasem.alsawadi.18@ucl.ac.uk
malswadi@kacst.edu.sa

2nd Miguel Rio
Department of Electronic and Electrical Engineering
University College London
London, UK
miguel.rio@ucl.ac.uk

***Abstract*—There has been a dramatic increase in the volume of videos and their related content uploaded to the internet. Accordingly, the need for efficient algorithms to analyse this vast amount of data has attracted significant research interest. This work aims to recognize activities of daily living using the ST-GCN model, providing a comparison between four different partitioning strategies: spatial configuration partitioning, full distance split, connection split, and index split. To achieve this aim, we present the first implementation of the ST-GCN framework upon the HMDB-51 dataset. Additionally, we show that our proposals have achieved the highest accuracy performance on the UCF-101 dataset using the ST-GCN framework than the state-of-the-art approach.**

***Index Terms*—Spatial Temporal Graph Convolution Network, Skeleton, HMDB-51, UCF-101, Action Recognition, Graph Neural Network**

I. Introduction

Recently, the amount of videos uploaded to the internet has increased substantially. According to Statista [1], by May 2019, more than 500 hours of video were uploaded to YouTube every minute, and the numbers did not slow down. Therefore, the need for robust algorithms to analyse this enormous amount of data has increased accordingly.

An action recognition system based upon human body motions is an efficient way of interpreting videos' contents. The skeleton movements approach offers multiple advantages over the other solutions. The skeleton information is robust to changes in the illumination of the environment where the action takes place. Also, it is robust to changes in the background [2]. For these reasons, we have chosen this approach to define the premise of our proposed method.

Our study is based upon the proposal presented by Yan *et al.* in [3]. Instead of analysing the frames of a video by their pixel values (i.e., RGB images), the authors first represent the actors as a set of joint-like keypoints using the OpenPose library [4]. To achieve the action recognition, the authors proposed the Spatial-Temporal Graph Convolutional Neural Network (ST-GCN) model. As the name indicates, this framework can analyse both the spatial and the temporal relations between the set of nodes (i.e., the skeleton joints) during the performance of the action. Subsequently, the model is trained in an end-to-end manner using a Graph Convolutional Neural Network (GCN) architecture [5].

Presently, there are multiple datasets available for research on human action recognition. Among these alternatives, the UCF-101 [6], and the HMDB-51 [7] datasets are considered to be reference benchmarks.

The ST-GCN model has been applied upon the UCF-101 dataset in order to recognize human actions in [8]. However, to the best of our knowledge, there is no previous evidence of using the ST-GCN model on the HMDB-51 dataset for action recognition. Therefore, in this study we present the first ST-GCN model implementation for the HMDB-51 dataset for action recognition. In what follows, we summarise our contributions below:

- We present the first results of the ST-GCN model trained on the HMDB-51 dataset for action recognition. In order to achieve that aim, we have performed a skeleton information extraction of both the UCF-101 and the HMDB-51 datasets.
- We have implemented our proposed partitioning strategies on the ST-GCN model [9] on the benchmark datasets (UCF-101 and the HMDB-51 datasets).
- We provide a deep analysis of the impact of different batch sizes during training upon the accuracy performance of the output models using the both benchmark datasets.
- Additionally, we have provided the open-source skeleton information of the UCF-101 and HMDB-51 datasets for the research community[1].

The remainder of the paper is structured as follows: in **Section II**, we present the state-of-the-art partitioning strategies for the ST-GCN model applied for action recognition. In **Section III** we explain the constraints we have used in our experiments.

[1] https://github.com/malswadi/skeleton-ucf-hmdb

The experimental results are described in depth in **Section IV**. Finally, **Section V** presents the summary and discussions.

II. Action recognition using ST-GCN

In order to perform the convolution operation, Yan *et al.* [3] first divided the skeleton into subsets of joints (*i.e., neighbor-sets*). Each of these sets are composed by a *root node* and its adjacent nodes (Fig. 1). On the other hand, each kernel has a size of K x K. For graphs with no specific order, priority criteria must be set in each neighbor-set to map each joint to a label. Hence, the convolution process can be performed, and network training can be possible.

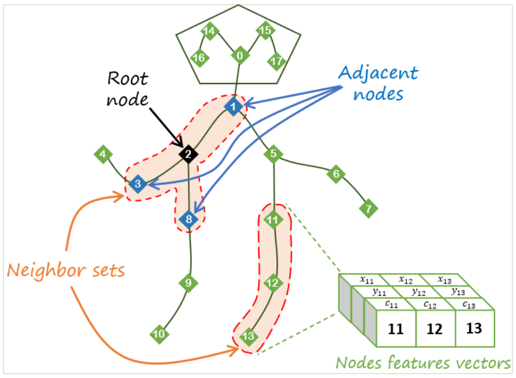

Fig. 1. Skeleton components and the keypoints indexes of OpenPose layout.

Partitioning strategies

A set of partitioning strategies were presented in [3]. Among these, the *spatial configuration partitioning* strategy offered the best results. We were able to improve this performance with the use of the ST-GCN model by presenting a novel set of partitioning strategies in [9]: the *full distance*, *connection* and *index splits*. All these strategies are going to be explained in the following sections.

1) Spatial Configuration: In this approach, the kernel size K = 3 and the center of gravity of the skeleton (average of the values on each joint axis across all the training set) is considered. Mathematically, the mapping for this strategy is defined with the following equation

$$_{ti}(v_{tj}) = \begin{cases} 0 & if \quad r_j = r_i \\ 1 & if \quad r_j < r_i \\ 2 & if \quad r_j > r_i \end{cases} , \tag{1}$$

where $l_{ti}(v_{tj})$ represents the label mapping for the node v_{tj}, r_j is the average value from the root node to the center of gravity and r_i is the average value from the i_{th} node to the center of gravity. As can be noticed, two nodes can share a single label within a neighbor set.

2) Full distance split: In this strategy, we considered the distance from *every joint* in the neighbor set to the center of gravity of the skeleton. Hence, each node in the neighbor set has an individual label. The nearest the node is to the center of gravity, the highest priority it is assigned to it [9] (Fig. 2.i)

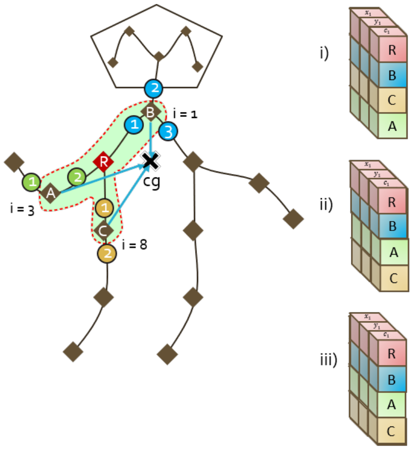

Fig. 2. Feature vector output of split strategies in [9]. i) Full distance. ii) Connection. iii) Index.

To describe this strategy mathematically, a set $\mathcal{F}$ is defined. This set contains the Euclidean distances of the i-th adjacent node u_{ti} (of the root node u_{tj}) with respect to the center of gravity of the skeleton, sorted in ascending order. With this auxiliary set in place, the label mapping can be defined using the Eq. 2.

$$_{ti}(u_{tj}) = \begin{cases} 0 & if \quad |u_{ti} - cg|_2 = x_r \\ m & if \quad |u_{ti} - cg|_2 = f_m \end{cases} , \tag{2}$$

where l_{ti} represents the label map for each joint u_{ti} in the neighbor set of the root node u_{tj}, x_r is the Euclidean distance from the root node u_{tj} to the center of gravity of the skeleton and f_m is the value of $\mathcal{F}$ at the m index.

3) Connection split: For this partitioning criteria, the degree of each vertex (i.e., the joints) of the skeleton graph is considered. The higher degree, the higher priority [9]. If several nodes share the same degree, their priority is set randomly (Fig. 2.ii).

To define the label mapping, we proposed a set $\mathcal{C}$ as the degree values of each of the N adjacent nodes of the root node sorted in descending order. Given the set $\mathcal{C}$ defined, the label mapping can be obtained using Eq. 3.

$$_{ti}(u_{tj}) = \begin{cases} 0 & if \quad d(u_{ti}) = d_r \\ m & if \quad d(u_{ti}) = c_m \end{cases} , \tag{3}$$

where l_{ti} represents the label map for each joint u_{ti} in the neighbor set of the root node u_{tj}, d_r is the degree corresponding the root node and c_m is the value of $\mathcal{C}$ at the m index.

4) Index split: For this strategy, we considered the OpenPose [4] output keypoints shown in Fig. 1. The smallest value of the keypoint index, the highest priority [9] (Fig. 2.iii). We defined an auxiliary set $\mathcal{P}$ with the key point index values of the adjacent nodes sorted in ascendant order. Then, the label mapping is obtained using Eq. 4.

$$_{ti}(u_{tj}) = \begin{cases} 0 & if \quad ind(u_{ti}) = in_r \\ m & if \quad ind(u_{ti}) = p_m \end{cases} , \tag{4}$$

where l_{ti} and $ind(u_{ti})$ represent the label map and the index keypoint value of the i_{th} joint, respectively; in_r is the index of the keypoint corresponding to the root node u_{tj} and p_m is the value of $\mathcal{P}$ at the m index.

III. Experimental settings

Given that the skeleton representation of the actors is not provided for either the UCF-101 [6] or the HMDB-51 [7], we first extract that skeleton representation from both datasets.

A. Skeleton Extraction

We followed the experiment guidelines provided in [3]. First, the resolution of each video sample has been resized into a fixed dimension of 340 × 256 pixels. Due to the variability of the duration of each clip, a fixed duration of 300 frames has been proposed. Therefore, if any video clip has less than 300 frames, we repeat the initial frames until we reach the amount needed. Otherwise, if the video clip exceeds the frame number, we trim it.

Third, we extracted the skeleton data with the use of the OpenPose library [4]. This system output the 2D coordinates of 18 main skeleton joints positions (shown in Fig. 1). Each skeleton joint information consists of three values as (x, y, c), where x and y are the cartesian coordinates in the horizontal and vertical axis, respectively, and c represents the confidence score of the detected joint. Hence, the Spatio-temporal information of the skeleton of each video sample is represented as a tensor with shape (18, 3, 300). By setting the T value to 300, our output tensor is illustrated in Fig. 3.

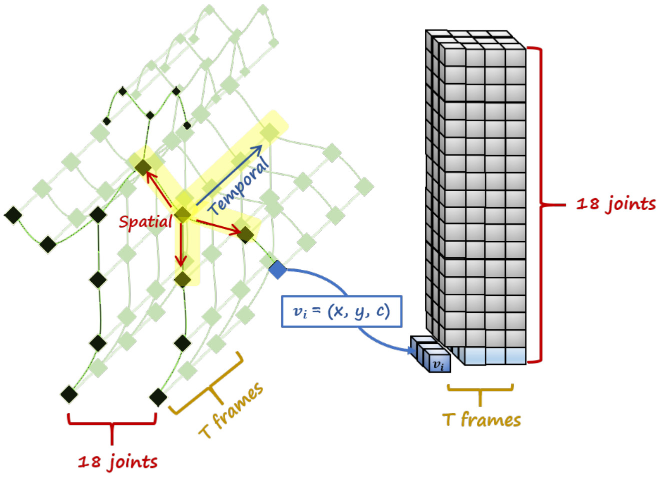

Fig. 3. Skeleton spatial-temporal data represented as a tensor.

Finally, we have iterated through the video clips of the datasets and saved the skeleton information as JSON files. These files are publicly available for the research community in [1] .

B. Datasets

The experiments were performed on 2 of the top 5 most popular datasets for action recognition [10]: the UCF-101 [6] and HMDB-51 [7] datasets.

1) UCF-101: The UCF-101 is the most commonly used benchmark human action dataset. Every video sample from this dataset is sourced from YouTube. The clip's duration varies from 1.06 sec to 71.04 sec and has a fixed frame rate of 25 fps and a fixed resolution of 320×240 pixels. This dataset provides a total of 13,320 clips classified into 101 action classes. These classes can be broadly divided into five major subsets: *Human-Object Interaction*, *Body-Motion Only*, *Human-Human Interaction*, *Playing Musical Instruments* and *Sports* [6].

2) HMDB-51: Aside from YouTube, the HMDB-51 dataset was collected from a wider range of sources (i.e., movies, Google videos, etc.). Due to the variability of resolution, the height of all the samples was scaled to 240 pixels, and the width has was scaled to maintain the original video ratio. Furthermore, the frame rate was modified to have a fixed value of 30 fps. It provides a total of 6,766 video clips of 51 different classes. These classes can be broadly classified into 5 categories: *General facial actions*, *Facial actions with object manipulation*, *General body movements*, *Body movements with object interaction* and *Body movements for human interaction* [7].

C. ST-GCN additional layer: the M-Mask

Not all the joints provide the same quality and quantity of information regarding the movement performed. In the ST-GCN framework, the authors added a mask M (or M-mask) to each layer of the GCN to express the importance of each joint [3]. According to their results, the proposed M-mask considerably improves their architecture's performance. Therefore, the authors constantly apply it to the ST-GCN network in their experiments.

D. Training Details

We performed the experiments of the implementation of the ST-GCN model using the spatial configuration partitioning [3] and the enhanced split strategies proposed in [9] to find the partitioning approach that offered the best performance in terms of accuracy. We also included experiments with and without the M-mask layer implementation to provide a valid comparison.

Every model has been trained for 80 epochs using the stochastic gradient descent with learning rate decay as optimization algorithm. The learning rate decays by a factor of 0.1 every 10^{th} epoch, starting from the epoch number 20.

Another experiment setting criteria was to find the optimal batch size. This hyperparameter allows the model to adjust its parameters during optimization with respect to a small subset of training samples called *mini batches* [11]. Therefore, we proposed this hyper-parameter to be one of the experiment's definition criteria. We vary the batch size from 8, 16, 32, 64, and 128.

IV. Results

The results correspond to the models with the best performance in terms of accuracy. In Table I are shown the results of these experiments. The accuracy values shown were obtained using top-1 criteria. The ”Y” (”Yes”) and ”N” (”No”) values in the ”M-mask” column represent whether the M-mask layer was implemented or not in that experiment, respectively. We have evaluated the model for each of the five epochs.

TABLE I
EXPERIMENTS PERFORMANCE
(TOP-1 ACCURACY)

Method	Batch size	M-mask	UCF-101	HMDB-51
Spatial C.P.	8	Y	46.42%	37.34%
		N	65.36%	40.77%
	16	Y	68.71%	44.39%
		N	65.89%	41.08%
	32	Y	68.96%	43.89%
		N	68.55%	45.45%
	64	Y	70.47%	45.64%
		N	68.18%	**46.82%**
	128	N	**70.72%**	44.64%
Full Distance Split	8	Y	48.51%	41.77%
		N	58.73%	33.23%
	16	Y	61.02%	38.97%
		N	67.89%	42.08%
	32	Y	69.16%	33.23%
		N	66.30%	45.51%
	64	Y	**70.43%**	42.02%
		N	68.59%	**45.82%**
	128	N	66.91%	45.26%
Connection Split	8	Y	63.03%	23.63%
		N	63.48%	39.34%
	16	Y	64.46%	43.27%
		N	62.99%	40.84%
	32	Y	70.88%	40.52%
		N	69.41%	41.52%
	64	Y	**70.96%**	32.29%
		N	68.18%	47.19%
	128	N	70.35%	**48.88%**
Index Split	8	Y	56.61%	38.97%
		N	58.24%	34.91%
	16	Y	69.33%	35.47%
		N	62.70%	46.57%
	32	Y	68.34%	43.20%
		N	68.34%	45.51%
	64	Y	72.31%	**47.69%**
		N	72.19%	43.39%
	128	N	**73.25%**	46.51%

By analysing the output values of the experiments shown in Table I, it can be noticed that, in most experiments, the output model tends to be more robust as the batch size increases.

A. UCF-101

To provide a valid comparison, we have also included the outcome of the previous ST-GCN implementation performed by Zheng *et al.* [8] in Table II.

The model with M-mask implementation that achieved the best accuracy performance was trained using a 64 batch size and utilized the index split partitioning strategy. It has achieved 1.84% of accuracy improvement with respect to the spatial configuration partitioning approach proposed by Yan *et al.* in [3]. Moreover, this model enhances the previous state-of-the-art results in [8] by 21.78%.

On the other hand, the index split partitioning strategy allowed the ST-GCN model to achieve the best accuracy performance with no use of the M-mask implementation. This solution enhanced the spatial configuration partitioning model approach proposed by Yan *et al.* in [3] by 2.53%.

TABLE II
UCF-101 PERFORMANCE USING M-MASK

	Method	Accuracy
ST-GCN	Spatial Configuration Partitioning	70.47%
Zheng *et al.* [8]	Spatial Configuration Partitioning	50.53%
Alsawadi and Rio [9]	Full Distance Split	70.43%
Alsawadi and Rio [9]	Connection Split	70.96%
Alsawadi and Rio [9]	**Index Split**	**72.31%**

In Figure 4 are shown 2 different examples of the action prediction achieved taken from the UCF-101 dataset. As it can be seen in the column (c), the ST-GCN model can output more than one label for a sample frame.

B. HMDB-51

There is no previous application of the ST-GCN model upon the HMDB-51 dataset to the author’s knowledge. Hence, the table only contains the results of the present study using the different partitioning strategies.

The highest value with M-mask implementation was achieved with the use of the index split partition strategy. This model was trained by choosing a training batch size of 64. It has reached more than 2% accuracy improvement with respect to the spatial configuration partitioning proposed by Yan *et al.* in [3].

In Figure 5 are shown 2 different examples of the action prediction achieved taken from the HMDB-51 dataset. As it can be noticed in the bottom example, the model is able to recognize the action of both of the actors accurately.

V. CONCLUSION

In this paper, we have proposed novel action recognition method using ST-GCN model by exploiting partitioning strategies: *spatial configuration paritioning*, *full distance split*, *connection split* and *index split*. We have presented the first implementation of the ST-GCN framework on the HMDB-51 [7] dataset achieving 48.88% top-1 accuracy by using the connection split partitioning approach. Our proposal outperforms the state-of-the-art using the ST-GCN framework on the UCF-101. Our results further show performance superiority over the most recent related work proposed in [9] with much lower training and computational inference costs and structural simplicity. However, the difference in the amount of training data impacted considerably in the final performance.

As future work, we propose increasing the size of nodes in the neighbor sets to capture the relationships between joints that are distant from each other, in order to increase the overall accuracy.

ACKNOWLEDGMENT

The authors acknowledge the support of King Abdulaziz City of Science and Technology (KACST).

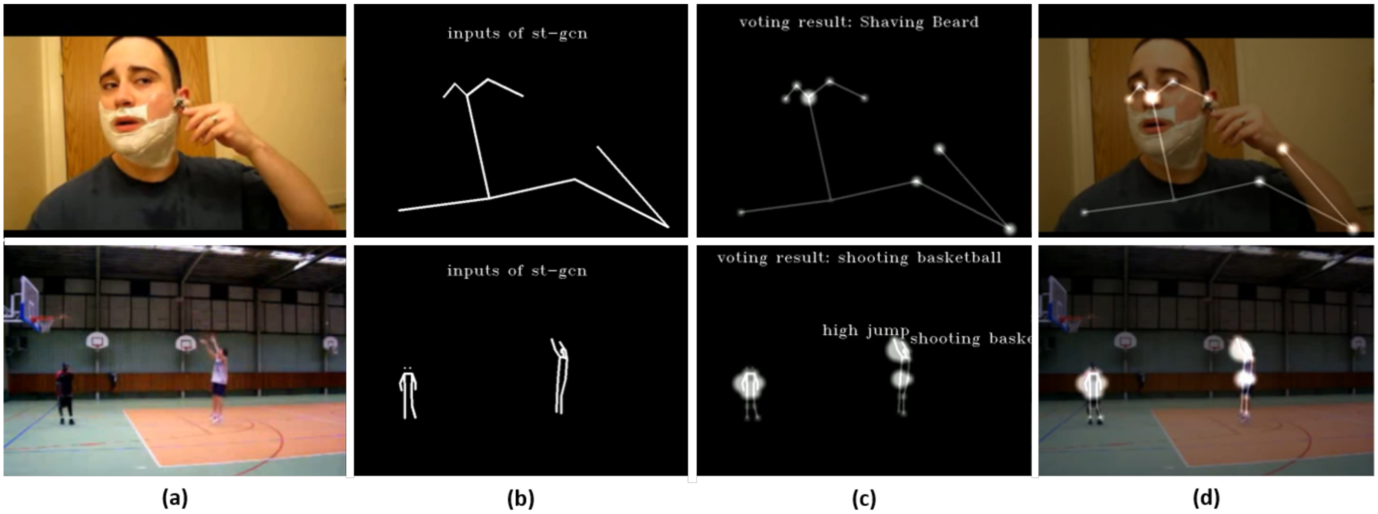

Fig. 4. UCF-101 Action prediction examples. (a) Original frame. (b) ST-GCN input (OpenPose estimation). (c) ST-GCN output attention and action prediction. (d) Original frame overlayed with ST-GCN output.

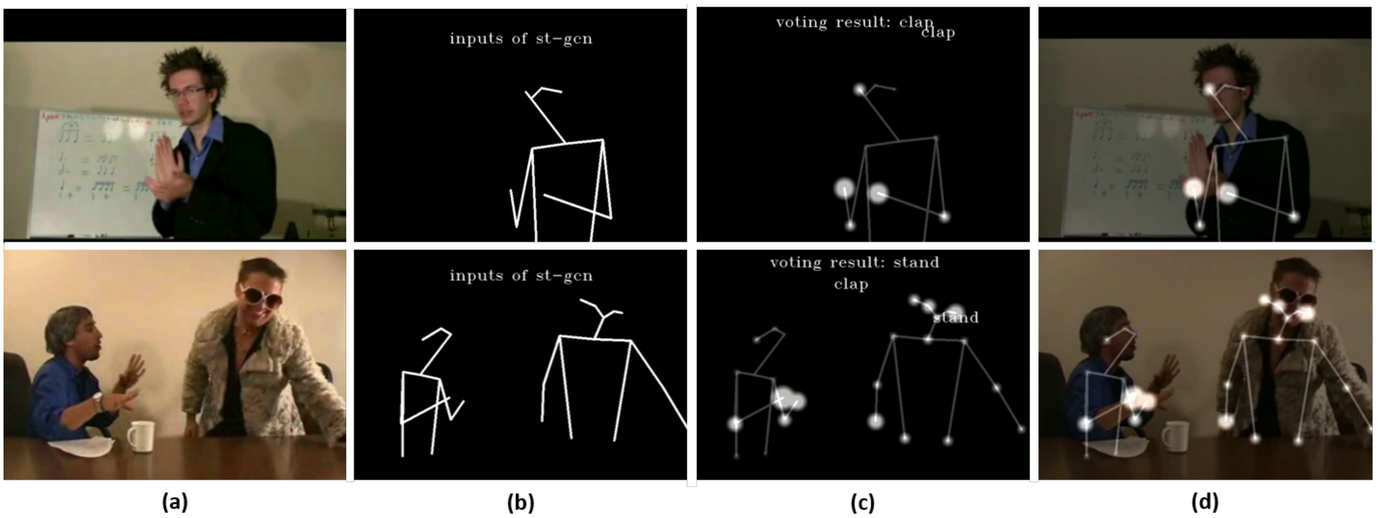

Fig. 5. HMDB-51 Action prediction examples. (a) Original frame. (b) ST-GCN input (OpenPose estimation). (c) ST-GCN output attention and action prediction. (d) Original frame overlayed with ST-GCN output.

References

[1] Statista Research Department, "YouTube: most subscribed channels 2021." [Online]. Available: https://www.statista.com/statistics/277758/most-popular-youtube-channels-ranked-by-subscribers/

[2] O. Keskes and R. Noumeir, "Vision-based fall detection using st-gcn," *IEEE Access*, vol. 9, pp. 28 224–28 236, 2021.

[3] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[4] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: realtime multi-person 2d pose estimation using part affinity fields," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 1, pp. 172–186, 2019.

[5] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[6] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

[7] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb: a large video database for human motion recognition," in *2011 International conference on computer vision*. IEEE, 2011, pp. 2556–2563.

[8] W. Zheng, P. Jing, and Q. Xu, "Action Recognition Based on Spatial Temporal Graph Convolutional Networks," in *Proceedings of the 3rd International Conference on Computer Science and Application Engineering*, 2019, pp. 1–5.

[9] M. S. Alsawadi and M. Rio, "Skeleton Split Strategies for Spatial Temporal Graph Convolution Networks," *arXiv preprint arXiv:2108.01309*, 2021.

[10] H.-B. Zhang, Y.-X. Zhang, B. Zhong, Q. Lei, L. Yang, J.-X. Du, and D.-S. Chen, "A Comprehensive Survey of Vision-Based Human Action Recognition Methods," *Sensors*, vol. 19, p. 1005, feb 2019.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.