How Much Context Matters? A Comparison for Skeleton-Based Activity Recognition

1st Matthias Tschöpe
Embedded Intelligence, DFKI
Kaiserslautern, Germany
matthias.tschoepe@dfki.de

2st Björn Friedrich
Embedded Intelligence, DFKI
Kaiserslautern, Germany
bjoern.friedrich@dfki.de

3st Sizhen Bian
Embedded Intelligence, DFKI
Kaiserslautern, Germany
sizhen.bian@dfki.de

4st Paul Lukowicz
Embedded Intelligence, DFKI
Kaiserslautern, Germany
paul.lukowicz@dfki.de

***Abstract*—Automated and anonymized detection of normal and unusual behavior can support clinical staff in hospitals and care facilities, where continuous observation of patients is often not possible. Such systems can help to improve the safety of patients and employees by detecting unusual or potentially critical situations while protecting privacy by avoiding the use of raw video data.**

In this work, we empirically compare models for classifying human activities using a 2D pose dataset recorded in a care-related context. We use five models and vary the temporal context by using six different sliding window configurations. In addition, we analyze how the use of a body-centered coordinate system changes the classification results. We evaluate all results using Leave-One-Subject-Out.

We focus on how temporal context, pose normalization, and the chosen models affect the classification results. The results show that graph-based and transformer-based models achieve similar classification results when sufficient temporal context is used. On our chosen dataset, the best classification results are achieved with ST-GCN by using a sliding window configuration of (180/90), we get an average accuracy of 79.65% and a macro F_1-Score of 79.86%. Finally, we provide the GPU usage and power consumption for each model.

***Index Terms*—Skeleton-based Human Activity Recognition, Body-Centered Coordinate System, ConvLSTM, MS-G3D, ST-GCN, TinierHAR, TinyHAR**

I. Introduction

Due to staff shortages and high workloads, it is often not possible to monitor patients or residents in hospitals or similar care facilities. Behavior causing harm to patients has a negative financial impact on the facilities. Extra cost of $4,617 per patient can incur [1]. Automated activity recognition systems can be helpful in such situations by alerting clinical staff when recognizing unusual or potentially dangerous activities. To ensure the subjects' anonymity, we only use approaches that use 2D skeleton data.

Previous works considered this problem in the ISAS Challenge 2025 on Unusual Activity Recognition [5]–[7]. These works show the importance of the task but also have some limitations: Most approaches evaluate only a small set of models, use a small set of sliding window sizes, and therefore cannot analyze how temporal context affects the performance, especially under Leave-One-Subject-Out (LOSO) evaluation.

In this work, we fill this gap by empirically comparing five different models to classify human activities based on 2D skeleton data. The models we considered are: ConvLSTM, ST-GCN, MS-G3D, TinyHAR, and TinierHAR. These models cover recurrent, graph-based, and small transformer-based approaches. Rather than introducing a new method, our goal is to analyze how different design choices affect the classification results.

In order to analyze the effects of temporal context, we evaluate each model using six sliding window configurations. We select sliding window sizes from one to six seconds to cover the typical duration of shorter and longer activities in the dataset. In addition, we analyze the impact of a body-centered coordinate system on the classification results. In addition to classification accuracy and macro F_1-Score, we provide the GPU memory usage and power consumption for each configuration to provide insights into the computational requirements, which can be relevant for the deployment on resource-constrained hardware. Thus, we can summarize our main contributions as:

- an empirical comparison of five models using LOSO,
- an analysis of temporal context using six sliding window configurations,
- an evaluation of body-centered pose normalization methods,
- a resource analysis considering GPU memory usage and power consumption.

The rest of this work is structured as follows: Section II provides an overview of related work on skeleton-based human activity recognition. Section III describes the dataset, preprocessing, and gives an overview of the models that we used. In Section IV we present our results and analyze them from different aspects. Finally, in Section V, we conclude our work.

II. Related Work

Early methods of skeleton-based human activity recognition focused mainly on recurrent neural networks. For example, in 2015, Du et al. [4] introduced a hierarchical RNN that divides the human body into meaningful parts and models their temporal movements in separate subnetworks, which are fused sequentially to capture increasingly complex motions. Later, Li et al. [10] proposed a two-stream CNN framework that directly processes raw joint coordinates and their temporal differences, supplemented by a skeleton transformer that automatically reorders and selects useful joints to improve recognition

performance. In 2016, Ordóñez et al. [14] introduced the ConvLSTM architecture, one of the first models to combine convolutional and recurrent layers for sequential sensor data.

When Yan et al. [25] introduced GCNs (Graph Convolutional Networks) in 2018, this led to a significant improvement in classification results. The idea was that the model developed by Yan et al. – known as ST-GCN – models spatial dependencies between joints using an adjacency matrix and temporal relationships using temporal convolutions. In 2020 Liu et al. [11] built up on this idea and introduced the MS-G3D model, which uses GCNs in combination with a multi-scale approach and a 3D spatio-temporal convolution to capture complex motion movements over varying time scales.

Transformer-based and attention-based methods have recently improved human activity recognition even more. Zhou et al. [26] introduced TinyHAR, which combines convolutional layers, attention layers, and an LSTM layer. TinyHAR was originally trained on IMU data, such as accelerometers, gyroscopes, and magnetometers. Based on this idea, Bian et al. [2] introduced TinierHAR in 2025, an even smaller model that achieves the same classification performance as TinyHAR on average.

Previous works also considered activity recognition based on 2D human pose data in clinical, healthcare, and nursing scenarios [8], [22]. In particular, previous studies have addressed tasks such as recognizing nursing activities, posture changes, or medical procedures using pose-based representations and learning-based models [9], [13], [15], [18], [19].

The aforementioned works provide an overview of the architectures which are often used in human activity recognition. These works cover a broad range of model architectures. However, there is no work yet that systematically analyzes which sliding window configurations are the best choice for these models on our selected dataset. Therefore, we select the five models: ConvLSTM as a recurrent neural network, ST-GCN and MS-G3D as graph neural networks, and TinyHAR as well as TinierHAR as modern attention-based neural networks. We compare them under the use of a LOSO evaluation over multiple sliding window configurations.

III. Dataset and Methods

A. Dataset

We use the dataset from Fujioka et al. [6], which was also used in the ISAS 2025 Unusual Activity Recognition Challenge [7]. The dataset contains recordings of human activities in hospitals, captured from five different subjects. Each subject performs normal and unusual activities, such as “Attacking”, “Biting nails”, “Eating snacks”, “Head banging”, “Sitting quietly”, “Throwing things”, “Using phone” and “Walking”. Fujioka et al. [6] used a GoPro 9 and an iPhone 15 Pro - both recorded with 30 FPS - to record the participants. Afterwards, they applied YOLOv7 [23] with its pose estimation model to extract the 2D human poses. The skeleton data consist of 17 joints per time step. The skeleton structure is shown in Figure 2.

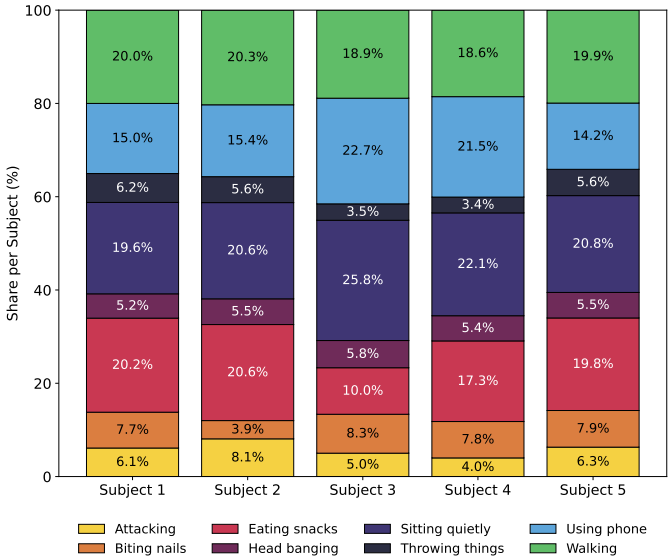

Fig. 1: This figure visualizes the per-subject class distribution of all classes in our used dataset at frame level.

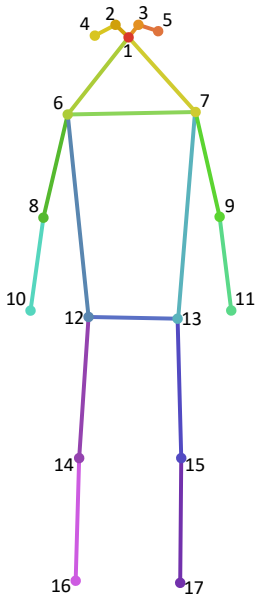

Fig. 2: Human pose structure that we used for our graph-based methods.

Figure 1 visualizes the per-subject class distribution in terms of the number of recorded frames per activity. For all subjects, activities such as “Walking”, “Sitting quietly”, and “Using phone” occur most often. On the other side, activities such as “Throwing things”, “Head banging”, and “Biting nails” occur less often.

B. Data Preprocessing

To be independent of the participants’ location, we use a body-centered coordinate system as proposed by Tschöpe et al. [20]. In the following, we briefly recap the idea of this transformation. The idea is to transform the (x, y) coordinates

into a new coordinate system, where all joints are relative to a reference point computed from a subset S of body joints.

As shown in Figure 2, the data set consists of 17 joints, where each joint is represented by its (x, y) coordinates. Thus, we have $F = 34$ features in our sliding windows. With this, let $\mathbf{W} \in \mathbb{R}^{T \times F}$ be a sliding window of length T. Further, we denote $\mathbf{W}_{t,:} = \left[x_1^{(t)}, y_1^{(t)}, \ldots, x_{17}^{(t)}, y_{17}^{(t)}\right]$ to refer to the t-th row in $\mathbf{W}$.

1) Body-centered coordinate system: Let ls_x be the column index in $\mathbf{W}$ of the x-coordinate from the left shoulder. Similarly, we define the column indices in $\mathbf{W}$ of x-coordinates from the right shoulder rs_x, the left hip lh_x, and the right hip rh_x. Analogously, we define the column indices of the y-coordinates from those joints. With this, we define the center-based origin of the body-centered coordinate system as follows:

$$x_{\text{center}}^{(t)} = \frac{1}{4} \sum_{j \in \mathcal{C}_x} \mathbf{W}_{t,j} \quad , \mathcal{C}_x = \{ls_x, rs_x, lh_x, rh_x\} \tag{1}$$

$$y_{\text{center}}^{(t)} = \frac{1}{4} \sum_{j \in \mathcal{C}_y} \mathbf{W}_{t,j} \quad , \mathcal{C}_y = \{ls_y, rs_y, lh_y, rh_y\} \tag{2}$$

The hip-based origin is then defined as:

$$x_{\text{hip}}^{(t)} = \frac{1}{2} \sum_{j \in \mathcal{C}_x} \mathbf{W}_{t,j} \quad \text{with } \mathcal{C}_x = \{lh_x, rh_x\} \tag{3}$$

$$y_{\text{hip}}^{(t)} = \frac{1}{2} \sum_{j \in \mathcal{C}_y} \mathbf{W}_{t,j} \quad \text{with } \mathcal{C}_y = \{lh_y, rh_y\} \tag{4}$$

Finally, we transform the coordinates from the pixel-based coordinate system to the body-centered coordinate system. Therefore, let $(o_x^{(t)}, o_y^{(t)})$ be either the center-based or hip-based origin. Each coordinate in $\mathbf{W}$ is then transformed by:

$$\widetilde{x}_k^{(t)} = x_k^{(t)} - o_x^{(t)}, \qquad \widetilde{y}_k^{(t)} = y_k^{(t)} - o_y^{(t)}. \tag{5}$$

With this, every keypoint left or above the chosen origin has negative values, while those to the right or below have positive values. This makes the movement independent of the position in the camera view. However, a limitation of this transformation is that it only normalizes translation and not body size. Therefore, variations in apparent skeleton size caused by different camera distances can cause camera-dependent effects.

C. Data Augmentation

Additionally, we randomly apply the following Data Augmentation methods: Dynamic Time Warping, Flipping, Gaussian Noise, Reverse, and Rotate. This set of data augmentation methods was inspired by the following previous works [3], [16], [17], [24].

D. Sliding Window Configurations

Since we want to analyze how much temporal context is necessary to get the best classification results, we consider the following six sliding window configurations:

- (30/15): Sliding window size of 30, step size of 15
- (60/30): Sliding window size of 60, step size of 30
- (90/45): Sliding window size of 90, step size of 45
- (120/60): Sliding window size of 120, step size of 60
- (150/75): Sliding window size of 150, step size of 75
- (180/90): Sliding window size of 180, step size of 90

As the cameras from the dataset recorded with 30 Hz, the sliding window contain temporal context from one to six seconds.

E. Model Architectures

We compare the following five models on the aforementioned dataset: ConvLSTM, ST-GCN, MS-G3D, TinyHAR, and TinierHAR. In the following, we briefly recap the model architectures and explain our adjustments.

a) ConvLSTM: The Convolutional Long Short-Term Memory Network (ConvLSTM) [14] combines convolutional layers for local feature extraction with recurrent layers to detect spatial and temporal dependencies. The network consists of two convolutional blocks, each with two 5×1 convolutions and ReLU activations, followed by temporal downsampling with a stride of 2. The extracted features are flattened and passed through a single LSTM layer.

b) ST-GCN: The Spatio-Temporal Graph Convolutional Network (ST-GCN) [25] represents humans as a graph, where each node corresponds to a body joint and edges follow the human pose of the skeleton. Spatial dependencies are modeled by graph convolutions, while temporal evolution is captured by 1D convolutions with kernel size 9 applied along the time axis. The network consists of ten spatio-temporal blocks with residual connections, batch normalization, and ReLU activations, progressively increasing channel dimensions from 64 to 256. Edge importance weighting parameters are learned for each layer.

c) MS-G3D: The Multi-Scale Graph 3D Network (MS-G3D) [11] extends the ST-GCN architecture with multi-scale graph kernels and 3D spatio-temporal convolutions. The network uses several temporal windows of different receptive field sizes (3, 5, 7) to jointly learn fine-grained and global motion dependencies. Liu et al [11] combined graph convolutions with multi-branch temporal convolutions using several dilation rates to capture short and long temporal patterns. Since the dataset only provides 2D skeleton coordinates, we adapt the original model by replacing the 3D spatio-temporal feature extraction with a 2D version that processes (x,y) keypoints instead of (x,y,z).

d) TinyHAR: TinyHAR [26] is a compact hybrid model combining convolutional, attention, and recurrent layers. The network starts with four consecutive 2D convolutional layers (3×1, 20 filters) with ReLU activations and batch normalization, which extract local temporal features and downsample the sequence length. A single self-attention block models interactions between input channels, followed by a fully connected fusion layer that reduces the feature dimensionality from 680 to 40, which integrates cross-channel information. A one-layer LSTM with a hidden size of 40 captures temporal dependencies, while a temporal attention mechanism emphasizes the most relevant time steps for classification.

e) TinierHAR: TinierHAR [2] is a further optimized lightweight variant of TinyHAR designed for embedded or real-time applications. It replaces standard convolutions with depthwise-separable convolutions and introduces residual shortcut connections to improve the gradient flow. The network consists of six convolutional blocks, in which the number of feature channels increases stepwise from one to four and finally to eight. These convolutional layers are followed by a bidirectional GRU with a hidden size of 16 that models temporal dependencies in both forward and backward directions. A simple attention layer then computes weights over the concatenated hidden states, producing a weighted temporal representation.

F. Training and Evaluation

We evaluate all models with all normalization approaches using LOSO. Thus, we use the data from four subjects for training and the data from the remaining subject for testing. Then we calculate the accuracy and macro F_1-Score for each test subject and calculate the mean and standard deviation of these metrics over all five test subjects. Further, we use a hyperparameter search to improve the following hyperparameters: Beta values for the AdamW optimizer, learning rate, γ value for learning rate scheduler, step size for learning rate scheduler, and weight decay. Furthermore, we use the AdamW optimizer [12] and train all models from scratch for 25 epochs. To compensate for the small number of trained epochs, we run the hyperparameter search for 300 individuals. Our hyperparameter search is a combination of random search and evolutionary search, adapted from these works [21]. In addition, we use class weights for the Cross-Entropy loss and assign higher class weights to underrepresented classes. Therefore we define for each class i, the class weight $w_i = \frac{\tilde{w}_i}{\sum_j \tilde{w}_j}$, where $\tilde{w}_i = (\frac{1}{c_i})^2$ and c_i denotes the total number of samples that belong to class i.

In addition, we use class weights in the Cross-Entropy loss to account for class imbalance. For each class, the weight is computed as the squared inverse of the total number of samples belonging to that class and then normalized across all classes so that the weights sum to one. This assigns higher importance to underrepresented classes during training while keeping the overall loss scale stable.

IV. Results and Discussion

A. Quantitative Results

Tables III–VIII show the classification results of all evaluated models with all three normalization approaches (baseline, center, and hip) for all six sliding window configurations. We provide the mean and standard deviation of the accuracy and macro F_1-Score over all five LOSO splits. In the following, we discuss the model performances for each sliding window configuration separately.

For the shortest sliding window configuration (30/15), TinyHAR (hip) achieves the best results with a macro F_1-Score of 68.89%, followed by TinierHAR (68.72%). With 67.71% ConvLSTM (center) is the third-best model. Afterwards, the graph-based models MS-G3D (center) and ST-GCN (center) follow with 66.49% and 65.81%, respectively.

Considering a window size of (60/30), ST-GCN (center) achieves the best macro F_1-Score with 72.34%. Interestingly, MS-G3D (center) - as our second graph-based model - performs for this sliding window configuration the worst, with only 63.55% macro F_1-Score. TinierHAR (center) also has a performance drop to 66.53% and is the second-worst model for this sliding window configuration. The second-best model is TinyHAR (center), followed by ConvLSTM (center) with 72.07% and 69.02%, respectively.

With a sliding window configuration of (90/45), TinyHAR (center) and ST-GCN (hip) are the only two models that achieve a macro F_1-Score higher than 75%, more exactly 77.15% and 76.98% respectively. MS-G3D (center) and TinierHAR (center) also perform well with 74.07% and 73.60% respectively. ConvLSTM (center) performs for this sliding window configuration the worst with a macro F_1-Score of 71.57%. From this sliding window configuration onward, the results of all models are more consistent, which shows that a temporal context of about 90 frames already cover most of the activity movements.

Looking at the models that achieve the best classification results on the sliding window configurations (120/60) and (150/75) while using the best normalization method for each model, then ST-GCN (center) is the best model, followed by TinyHAR (center). The remaining models change in order, but are all closely together, except for MS-G3D (hip) when considering a sliding window configuration of (150/75). For this sliding window configuration, the three best models are very close together 78.52% (ST-GCN), 78.47% (TinyHAR) and 77.16% (ST-GCN).

On the longest sliding window configuration (180/90), the best result was again achieved by ST-GCN (center) with a macro F_1-Score of 79.86%. The second-best model is TinyHAR (center) with a macro F_1-Score of 78.22%, which is slightly below compared to TinyHAR (center) on a sliding window configuration of (150/75). The third-best model is MS-G3D (center), followed by ConvLSTM (center) with 78.72% and 78.21%, respectively. Although TinierHAR (hip) improved compared to its performance on a sliding window configuration (150/75), its macro F_1-Score is 76.15%.

We performed the inference tests for power consumption and VRAM measurements on a Nvidia A100. Therefore, we used the measurements over all five LOSO splits and calculated the average. Thus, we list the average VRAM usage and standard deviation in the Tables III–VIII. The model with the highest power consumption is ST-GCN using a sliding window configuration of (150/75) with 83 Wh. On the other hand, MS-G3D (30/15) and TinyHAR (90/45) have the lowest power consumption at 73 Wh. The required VRAM ranges from 524 MB (TinierHAR (150/75)) to 995 MB (MS-G3D (180/90)). The effect of larger sliding window sizes is measurable only for a few models due to higher VRAM requirements, most clearly for MS-G3D. However, for practical applications, this is not critical, as all tested models require less than 1 GB of

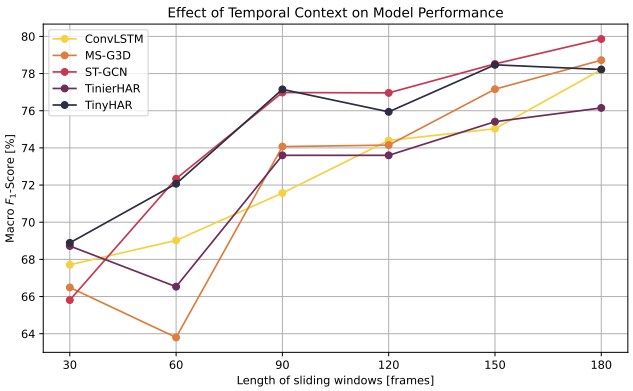

Fig. 3: This figure visualizes the effect of the temporal context based on the model performance in macro F_1-Score.

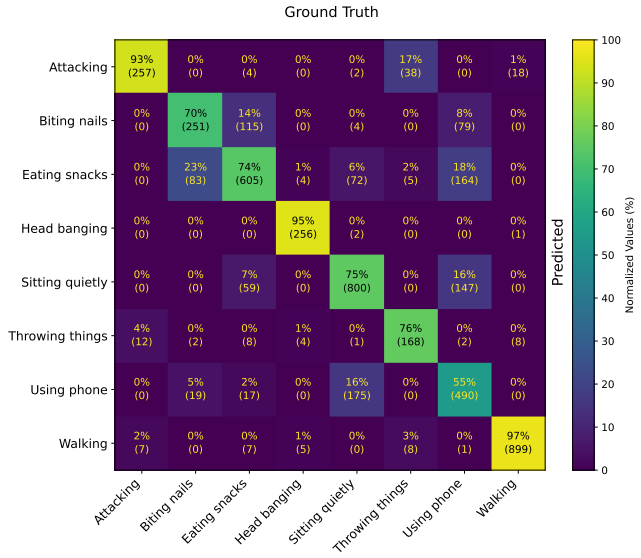

Fig. 4: Summed confusion matrix for ConvLSTM (center) with sliding window configuration (180/90).

VRAM and can therefore be applied even on older consumer hardware and some microcontrollers.

B. Influence of Pose Normalization

In Section IV-A, we discussed the best models per sliding window configuration using the normalization method with the best classification results. There, we noticed that the best-performing models always use the body-centered coordinate system. The largest improvement is given by ConvLSTM, especially when using larger sliding window sizes. For example, ConvLSTM (30/15) improves from 56.79% macro F_1-Score (baseline) to 67.04% macro F_1-Score (hip) and from 46.24% macro F_1-Score (baseline) to 78.21% macro F_1-Score (center) for the largest sliding window size. This improvement shows that using a body-centered coordinate system helps to remove unnecessary global translation and simplifies the movements consistent over all subjects. Between both body-centered coordinate systems, center-based normalization achieved slightly better overall results.

C. Impact of Temporal Context

Figure 3 shows how different sliding window sizes influence the macro F_1-Score for each model. For each sliding window size, we choose the best result over all normalization methods. Most models benefit from larger sliding window sizes, but the effect is not monotonic, especially for shorter sliding window sizes.

For small sliding windows (30–90 frames), the results vary within the models. MS-G3D and TinierHAR decrease from 30 to 60 frames (66.49% to 63.80% and 68.72% to 66.53%, respectively), whereas ST-GCN and TinyHAR already reach relatively good results at 60 frames (72.34% and 72.07%, respectively).

Looking at the results for sliding windows of size 90 and 120, we see that most models do not improve. However, if we use sliding window sizes above 120, most models improve more steadily, although the improvements are getting smaller. While ConvLSTM always improves by using a larger sliding window size, ST-GCN shows a nearly linear increase between 120 and 180 frames per sliding window. On the other hand, TinyHAR achieves its best performance with the second-largest sliding window size, which again shows that larger sliding window sizes are not always useful for all model architectures.

D. Model-based Results and Confusion Matrix Analysis

Overall sliding window configurations, ST-GCN (center) (180/90) achieves - with 79.86% - the highest macro F_1-Score, followed by MS-G3D (center) (180/90) with a macro F_1-Score of 78.72%.

To better understand the class-wise performance of the models, we also provide the confusion matrices with their best normalization method and sliding window configuration for each model in Figures 4-8. Over all models, the class “Walking” class is recognized most reliably with a true positive rate (TP) of over 95%. The classes “Attacking” and “Head banging” are also classified with TP rates of over 85% by all models except MS-G3D. Interestingly, MS-G3D performs best on the more static classes such as “Sitting quietly” and “Throwing things”, and achieves a TP rate of at least 82% for both classes.

Although ConvLSTM is not the best model overall, it performs very good for “Attacking”, “Head banging”, and “Walking”, with TP rates of at least 93% for each (see Figure 4). However, its performance is much lower for other classes. The class “Using phone” is particularly difficult and is often confused with “Eating snacks” or “Sitting quietly”. This is understandable, as several participants performed these activities while they were sitting, which results in visually similar poses.

Among all evaluated models, ST-GCN achieves the most balanced performance over all classes, and classifies correctly six out of eight classes with a TP rate above 79% (see

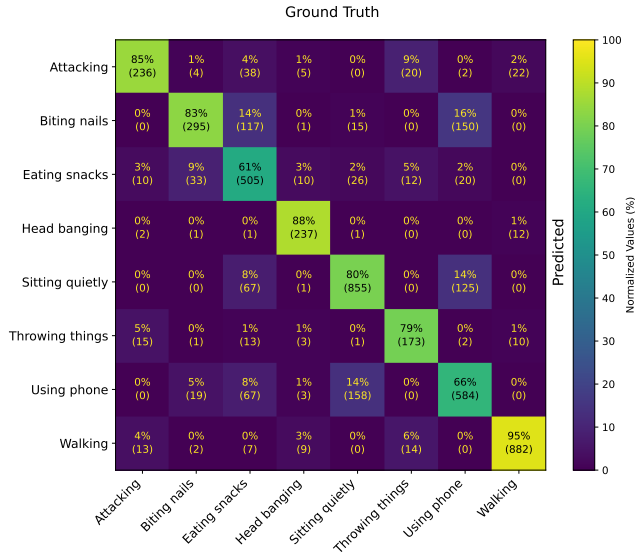

Fig. 5: Summed confusion matrix for ST-GCN (center) with sliding window configuration (180/90).

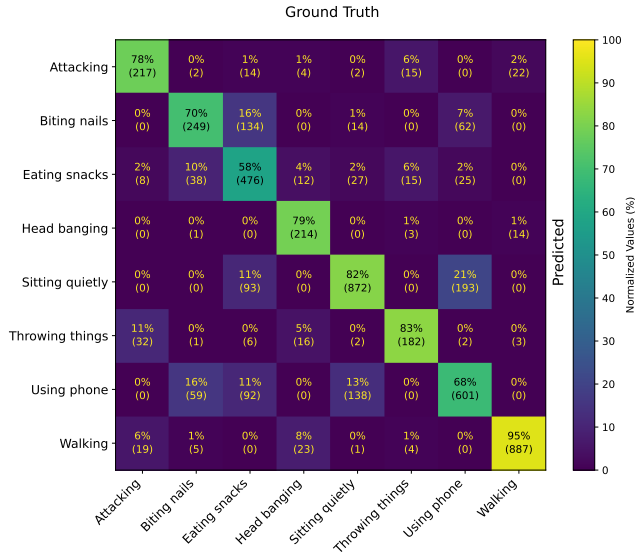

Fig. 6: Summed confusion matrix for MS-G3D (baseline) with sliding window configuration (180/90).

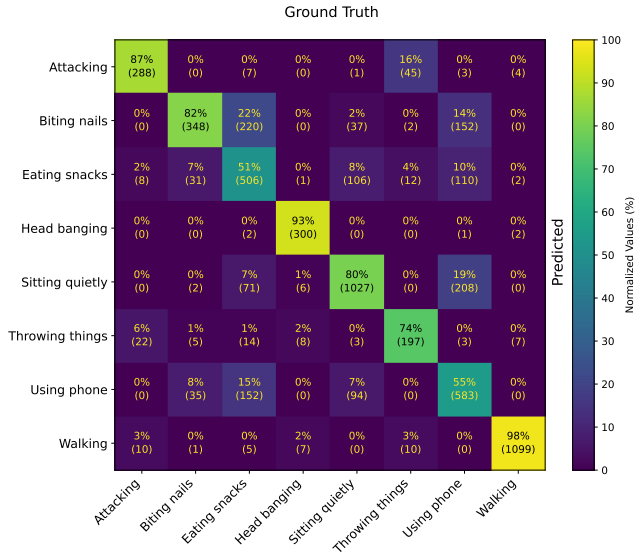

Fig. 7: Summed confusion matrix for TinierHAR (center) with sliding window configuration (150/75).

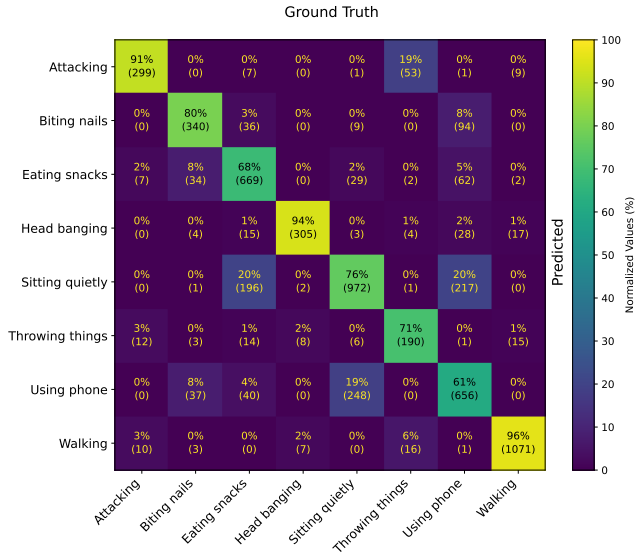

Fig. 8: Summed confusion matrix for TinyHAR (center) with sliding window configuration (150/75).

Figure 5). Only "Eating snacks" and "Using phone" are recognized less accurately. TinyHAR follows closely behind and achieves TP rates above 80% in five classes (see Figure 8), but in addition to the two classes mentioned above, it also has difficulties with "Throwing things" (74%). Although TinyHAR only achieves TP rates above 80% in four classes, it achieves low TP rates in the remaining four classes. For example, the class "Using phone" with the lowest TP rate still achieves 61%, while TinierHAR has two classes below 55% TP rate (see Figure 7).

Overall, the confusion matrices confirm the trends from the quantitative results: ST-GCN achieves the most stable and balanced predictions, while TinyHAR achieves comparable accuracy at lower computational cost. TinyHAR shows slightly higher variability at the class level but is still efficient and competitive. ConvLSTM performs exceptionally well with expressive motion patterns, but less well with complex or static activities.

E. Per-Subject Performance Analysis

In addition to the overall evaluation, we also analyze how the models perform for each individual test subject under the LOSO conditions. Figure 9 summarizes the macro F_1-Scores for all five models. The results show differences between subjects, which means that some individuals are more difficult to classify consistently than others.

Across all models, Subject 5 achieves the highest macro F_1-Scores, ranging from 88.36% (TinierHAR) to 91.80% (ST-

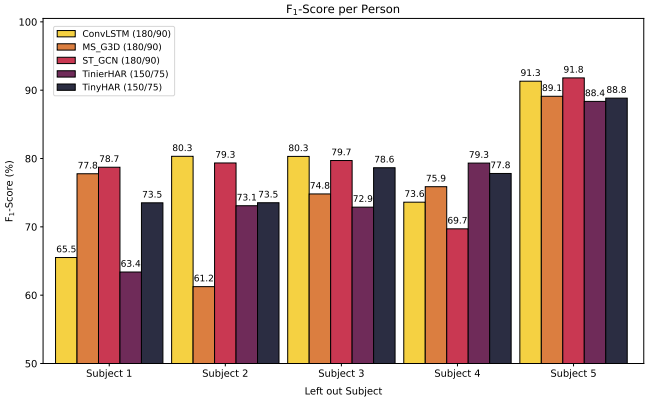

Fig. 9: Macro F_1-Scores per-subject for some selected models.

GCN). Subject 1 also shows high scores for most models, especially MS-G3D (77.77%) and ST-GCN (78.72%). In contrast, Subject 3 and Subject 4 tend to result in lower scores, particularly for ConvLSTM and TinierHAR. For example, ConvLSTM drops from 80.32% (Subject 2) to 73.61% (Subject 4), and TinierHAR decreases from 73.09% (Subject 2) to 72.88% (Subject 3).

ST-GCN shows the most stable performance over all subjects, with scores ranging from 69.71% to 91.80%, followed closely by TinyHAR, which remains between 73.52% and 88.84%. On the other side ConvLSTM shows the largest variations, spanning from 65.51% (Subject 1) to 91.31% (Subject 5). TinierHAR also shows strong subject dependence, reaching its best result on Subject 4 (79.33%) but considerably lower values on Subject 1 and 3.

Overall, these results suggest that all models capture the general structure of the activities, but their performance highly depends on how consistently a subject performs the movements. This shows the importance of preprocessing steps that reduce subject-specific differences.

F. Pairwise Statistical Comparison of Models

TABLE I: This table shows the test statistics W of the pairwise Wilcoxon signed-rank test computed from per-subject macro F_1-Score differences between the best model configurations.

Model	ConvLSTM	MS_G3D	ST_GCN	TinierHAR	TinyHAR
ConvLSTM	–	6	6	3	7
MS_G3D	6	–	4	7	5
ST_GCN	6	4	–	4	5
TinierHAR	3	7	4	–	3
TinyHAR	7	5	5	3	–

TABLE II: Analogously to Table I, this table shows the pairwise p-values of the Wilcoxon signed-rank test.

Model	ConvLSTM	MS_G3D	ST_GCN	TinierHAR	TinyHAR
ConvLSTM	–	0.4062	0.4062	0.1562	0.5938
MS_G3D	0.6875	–	0.8438	0.5000	0.7812
ST_GCN	0.6875	0.2188	–	0.2188	0.3125
TinierHAR	0.9062	0.5938	0.8438	–	0.9062
TinyHAR	0.5000	0.3125	0.7812	0.1562	–

To analyse whether the differences in the classification results between the models are statistically consistent over all subjects, we perform a pairwise statistical comparison using the Wilcoxon rank sum test. For each model, we first select the configuration with the best performance in terms of the macro F_1-Score, considering all combinations of normalization methods and sliding window configurations. Based on these best configurations, we calculate the macro F_1-Scores per subject and apply pairwise the Wilcoxon signed-rank tests between all model pairs by using the subject-related differences. As final results, we show the Wilcoxon test statistic $W = \min(W^+, W^-)$ and the related p - values.

Table I shows the Wilcoxon signed-rank test statistic W, which summarizes how consistently the classification results differ between pairs of models over all subjects. Lower values mean that a model tends to achieve more consistent classification results over all subjects. For example, the test statistic between ConvLSTM and TinierHAR is $W = 3$, which means that one of the models - in this case ConvLSTM - often, but not always, achieves better classification results.

Table II shows the corresponding p-values from the directed Wilcoxon tests. Unlike the W test statistics, this table is not symmetric, as each entry tests the hypothesis that the model listed in the row achieves higher classification results than the model listed in the column. For example, the p-value of 0.4062 for ConvLSTM versus ST-GCN reflects the probability of observing the measured subject-related differences (or more extreme differences) under the null hypothesis that ConvLSTM does not systematically achieve higher classification results than ST-GCN, while the reverse comparison has a p-value of 0.6875.

In summary, a few model pairs have relatively low W values, which indicates trends in differences in classification results, but none of the comparisons reach statistical significance ($p < 0.05$). This result suggests that the relative order of the models varies depending on the subject. Given the small number of subjects in the data set, this analysis should be interpreted as a complementary check rather than a definitive ranking. In this context, the statistical comparison supports the conclusions drawn from the quantitative results by highlighting that the observed differences are not dominated by a single subject but are instead influenced by variability between subjects.

V. Conclusion and Future Work

In this article, we evaluate 90 different model configurations trained from five different models (ConvLSTM, MS-G3D, ST-GCN, TinierHAR and TinyHAR), each with three different normalization methods (baseline, center and hip) and six different sliding window configurations. As we have seen in Section IV-C, for our specific dataset and tested models, sliding window sizes between three and six seconds are a good trade-off. Most of our tested models achieve even better results on a sliding window size of six seconds. In section IV-D we have shown that each model has specific strengths for different types of activities. The statistical comparison in Section IV-F

has shown that ConvLSTM is in four out of five subjects better than the more modern TinierHAR. However, ST-GCN (center) performs the best by using a sliding window size of six seconds. Finally, we evaluated the power consumption and VRAM usage, which showed that even the largest models with the largest sliding window configuration need less than 1 GB of VRAM, which makes it easy to apply them on consumer hardware for less than $1,000. Compared to the extra cost ($4,617) of a single harmed patient, this is much less and can be very helpful.

Our main focus for future work will be to develop a new model that combines the strengths of graph-based models and transformer-based models. Further, we want to improve the normalization method so that it also independent of the body-size. Furthermore, we want to extend our evaluation with our new model to additional datasets. In doing so, we also plan to create and annotate new datasets in the healthcare context ourselves in order to increase data diversity as we have seen that five subjects are too few to draw strong statistical conclusions. We also plan to evaluate the impact of different skeleton-data specific augmentation methods.

VI. Acknowledgements

The research reported in this paper was supported by the BMBF Cross-Act project (01IW25001).

References

[1] Lee Adler, David Yi, Michael Li, Barry McBroom, Loran Hauck, Christine Sammer, Cason Jones, Terry Shaw, and David Classen. Impact of inpatient harms on hospital finances and patient clinical outcomes. *Journal of Patient Safety*, 14:67–73, 6 2018.

[2] Sizhen Bian, Mengxi Liu, Vitor Fortes Rey, Daniel Geissler, and Paul Lukowicz. Tinierhar: Towards ultra-lightweight deep learning models for efficient human activity recognition on edge devices. In *Proceedings of the 2025 ACM International Symposium on Wearable Computers*, pages 163–169, 2025.

[3] Shou-Hsuan Chen, Hong-Rui Pan, and Shi-Yu Lai. Skeleton-based action recognition and evaluation using dynamic time warping algorithm enhanced by spatial-temporal feature engineering techniques. In *2024 International Conference on Consumer Electronics-Taiwan (ICCE-Taiwan)*, pages 795–796. IEEE, 2024.

[4] Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1110–1118, 2015.

[5] Taihei Fujioka, Christina Garcia, and Sozo Inoue. Challenge: Abnormal activity detection in individuals with developmental disabilities, 2025.

[6] Taihei Fujioka, Christina Garcia, and Sozo Inoue. Toward abnormal activity recognition of developmentally disabled individuals using pose estimation. *International Journal of Activity and Behavior Computing*, 2025(1):1–28, 2025.

[7] Christina Garcia, Nhat Tan Le, Taihei Fujioka, Umang Dobhal, Milyun Ni'ma Shoumi, Thanh Nha Nguyen, and Sozo Inoue. Summary of the unusual activity recognition challenge for developmental disability support. In *Proceedings of the International Symposium on Applied Science (ISAS 2025)*, 2026. to appear.

[8] Youssef Hbali, Sara Hbali, Lahoucine Ballihi, and Mohammed Sadgal. Skeleton-based human activity recognition for elderly monitoring systems. *IET Computer Vision*, 12(1):16–26, 2018.

[9] Samiul Islam, S. M. Hozaifa Hossain, Md. Zasim Uddin, Shahera Hossain, and Md Atiqur Rahman Ahad. Enhancing nursing activity recognition during endotracheal suctioning through video-based pose estimation and machine learning. *International Journal of Activity and Behavior Computing*, 2024(3):1–15, 2024.

[10] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. Skeleton-based action recognition with convolutional neural networks. In *2017 IEEE international conference on multimedia & expo workshops (ICMEW)*, pages 597–600. IEEE, 2017.

[11] Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, and Wanli Ouyang. Disentangling and unifying graph convolutions for skeleton-based action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 143–152, 2020.

[12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[13] Hoang Anh Vy Ngo, Quynh N Phuong Vu, Noriyo Colley, Shinji Ninomiya, Satoshi Kanai, Shunsuke Komizunai, Atsushi Konno, Misuzu Nakamura, and Sozo Inoue. Toward recognizing nursing activity in endotracheal suctioning using video-based pose estimation. *International Journal of Activity and Behavior Computing*, 2024(1):1, 2024.

[14] Francisco Javier Ordóñez and Daniel Roggen. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115, 2016.

[15] Hoang Khang Phan, Tu Nhat Khang Nguyen, Truong Vi Bui, Khuong Cong Duy Nguyen, Tuan Phong Nguyen, and Nhat Tan Le. Recognition of endotracheal suctioning activities: A feature extraction and ensemble learning approach based on pose estimation data. In *2024 International Conference on Activity and Behavior Computing (ABC)*, pages 1–9. IEEE, 2024.

[16] Haocong Rao, Shihao Xu, Xiping Hu, Jun Cheng, and Bin Hu. Augmented skeleton based contrastive action learning with momentum lstm for unsupervised action recognition. *Information Sciences*, 569:90–109, 2021.

[17] Samsu Sempena, Nur Ulfa Maulidevi, and Peb Ruswono Aryan. Human action recognition using dynamic time warping. In *Proceedings of the 2011 international conference on electrical engineering and informatics*, pages 1–5. IEEE, 2011.

[18] Sonja Stabenow, Lars Wagner, Alois Knoll, Klaus Bengler, and Dirk Wilhelm. Action recognition in medical environments for robotic assistance. *International Journal of Computer Assisted Radiology and Surgery*, pages 1–12, 2025.

[19] Matthias Tschöpe, Stefan Gerd Fritsch, David Habusch, Vitor Fortes Rey, Agnes Grünerbl, and Paul Lukowicz. Evaluating deep learning models for posture and movement recognition during the abcde protocol in nurse education. In *2024 International Conference on Activity and Behavior Computing (ABC)*, pages 1–10. IEEE, 2024.

[20] Matthias Tschöpe, Kirsten Harms, Daniel Eckhoff, Andreas Hein, Paul Lukowicz, and Bjorn Friedrich. Unusual activity recognition based on 2d-skeleton data. 2025. Accepted at the ISAS 2025 Challenge; to appear in the SCOPUS-indexed Journal of ISAS.

[21] Matthias Tschöpe, Kirsten Harms, Daniel Eckhoff, Paul Lukowicz, Andreas Hein, and Bjorn Friedrich. A data-centric approach to human activity recognition: Enhancing industrial har with virtual data generation. *International Journal of Activity and Behavior Computing*, 2025(2):1–16, 2025.

[22] Matthias Tschöpe, Dennis Schneider, Sungho Suh, and Paul Lukowicz. A novel guidance framework for nasal rapid antigen tests with improved swab keypoint detection. *Smart Health*, 35:100534, 2025.

[23] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7464–7475, 2023.

[24] Chu Xin, Seokhwan Kim, Yongjoo Cho, and Kyoung Shin Park. Enhancing human action recognition with 3d skeleton data: A comprehensive study of deep learning and data augmentation. *Electronics*, 13(4):747, 2024.

[25] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[26] Yexu Zhou, Haibin Zhao, Yiran Huang, Till Riedel, Michael Hefenbrock, and Michael Beigl. Tinyhar: A lightweight deep learning model designed for human activity recognition. In *Proceedings of the 2022 ACM International Symposium on Wearable Computers*, pages 89–93, 2022.

TABLE III: Results for sliding window configuration (30/15)

Approach	**Accuracy**	**macro F_1-Score**	**GPU [Wh]**	**VRAM [MB]**
ConvLSTM (baseline)	$60.21\% \pm 3.29\%$	$56.79\% \pm 6.88\%$	76	551 ± 69
ConvLSTM (center)	$66.17\% \pm 6.43\%$	$67.71\% \pm 8.33\%$	76	551 ± 69
ConvLSTM (hip)	$68.35\% \pm 4.76\%$	$67.04\% \pm 7.15\%$	76	551 ± 69
MS-G3D (baseline)	$64.46\% \pm 9.03\%$	$63.83\% \pm 9.43\%$	73	636 ± 20
MS-G3D (center)	$65.38\% \pm 6.51\%$	$66.49\% \pm 10.25\%$	73	636 ± 20
MS-G3D (hip)	$58.47\% \pm 6.74\%$	$58.91\% \pm 6.43\%$	73	636 ± 20
ST-GCN (baseline)	$66.72\% \pm 8.61\%$	$62.11\% \pm 10.62\%$	82	561 ± 35
ST-GCN (center)	$66.59\% \pm 7.06\%$	$65.81\% \pm 9.89\%$	82	561 ± 35
ST-GCN (hip)	$63.45\% \pm 8.61\%$	$62.88\% \pm 9.58\%$	82	561 ± 35
TinierHAR (baseline)	$50.88\% \pm 10.98\%$	$46.25\% \pm 14.96\%$	73	546 ± 49
TinierHAR (center)	$70.22\% \pm 3.93\%$	$68.72\% \pm 6.43\%$	73	546 ± 49
TinierHAR (hip)	$64.49\% \pm 15.33\%$	$65.52\% \pm 16.37\%$	73	546 ± 49
TinyHAR (baseline)	$63.78\% \pm 4.25\%$	$58.61\% \pm 6.30\%$	82	533 ± 33
TinyHAR (center)	$69.82\% \pm 5.77\%$	$67.38\% \pm 6.83\%$	82	533 ± 33
TinyHAR (hip)	**$71.37\% \pm 7.29\%$**	**$68.89\% \pm 9.32\%$**	82	533 ± 33

TABLE IV: Results for sliding window configuration (60/30)

Approach	**Accuracy**	**macro F_1-Score**	**GPU [Wh]**	**VRAM [MB]**
ConvLSTM (baseline)	$62.40\% \pm 3.02\%$	$55.50\% \pm 3.55\%$	76	545 ± 94
ConvLSTM (center)	$68.70\% \pm 7.82\%$	$69.02\% \pm 3.59\%$	76	545 ± 94
ConvLSTM (hip)	$68.11\% \pm 3.46\%$	$66.89\% \pm 3.97\%$	76	545 ± 94
MS-G3D (baseline)	$62.03\% \pm 8.41\%$	$60.94\% \pm 10.19\%$	82	704 ± 32
MS-G3D (center)	$64.25\% \pm 14.53\%$	$63.55\% \pm 13.96\%$	82	704 ± 32
MS-G3D (hip)	$63.76\% \pm 8.17\%$	$63.80\% \pm 7.14\%$	82	704 ± 32
ST-GCN (baseline)	$67.64\% \pm 8.08\%$	$68.16\% \pm 6.58\%$	78	589 ± 53
ST-GCN (center)	$71.00\% \pm 9.29\%$	**$72.34\% \pm 9.57\%$**	78	589 ± 53
ST-GCN (hip)	**$73.45\% \pm 3.34\%$**	$71.12\% \pm 2.79\%$	78	589 ± 53
TinierHAR (baseline)	$62.75\% \pm 9.31\%$	$58.60\% \pm 14.09\%$	77	534 ± 93
TinierHAR (center)	$67.02\% \pm 10.35\%$	$66.53\% \pm 10.98\%$	77	534 ± 93
TinierHAR (hip)	$62.18\% \pm 12.85\%$	$64.35\% \pm 13.01\%$	77	534 ± 93
TinyHAR (baseline)	$68.81\% \pm 7.13\%$	$63.03\% \pm 7.95\%$	77	532 ± 51
TinyHAR (center)	$72.12\% \pm 4.26\%$	$72.07\% \pm 5.79\%$	77	532 ± 51
TinyHAR (hip)	$73.34\% \pm 7.60\%$	$70.06\% \pm 12.63\%$	77	532 ± 51

TABLE V: Results for sliding window configuration (90/45)

Approach	**Accuracy**	**macro F_1-Score**	**GPU [Wh]**	**VRAM [MB]**
ConvLSTM (baseline)	$57.31\% \pm 10.01\%$	$57.50\% \pm 7.70\%$	77	529 ± 121
ConvLSTM (center)	$73.04\% \pm 5.20\%$	$71.57\% \pm 7.35\%$	77	529 ± 121
ConvLSTM (hip)	$70.41\% \pm 7.61\%$	$69.31\% \pm 10.87\%$	77	529 ± 121
MS-G3D (baseline)	$73.64\% \pm 11.58\%$	$72.34\% \pm 12.70\%$	78	766 ± 42
MS-G3D (center)	$70.92\% \pm 5.35\%$	$74.07\% \pm 4.96\%$	78	766 ± 42
MS-G3D (hip)	$73.51\% \pm 5.83\%$	$74.05\% \pm 4.22\%$	78	766 ± 42
ST-GCN (baseline)	$75.88\% \pm 7.12\%$	$73.02\% \pm 9.95\%$	81	612 ± 62
ST-GCN (center)	$74.06\% \pm 5.22\%$	$74.88\% \pm 6.77\%$	81	612 ± 62
ST-GCN (hip)	$76.24\% \pm 4.35\%$	$76.98\% \pm 5.34\%$	81	612 ± 62
TinierHAR (baseline)	$63.42\% \pm 10.38\%$	$58.46\% \pm 11.98\%$	76	539 ± 86
TinierHAR (center)	$75.12\% \pm 3.24\%$	$73.60\% \pm 5.67\%$	76	539 ± 86
TinierHAR (hip)	$71.03\% \pm 6.62\%$	$70.79\% \pm 8.11\%$	76	539 ± 86
TinyHAR (baseline)	$66.81\% \pm 5.64\%$	$64.09\% \pm 5.00\%$	73	532 ± 50
TinyHAR (center)	**$77.22\% \pm 5.17\%$**	**$77.15\% \pm 6.94\%$**	73	532 ± 50
TinyHAR (hip)	$70.98\% \pm 6.41\%$	$70.80\% \pm 7.28\%$	73	532 ± 50

TABLE VI: Results for sliding window configuration (120/60)

Approach	Accuracy	macro F_1-Score	GPU [Wh]	VRAM [MB]
ConvLSTM (baseline)	$62.76\% \pm 4.78\%$	$58.04\% \pm 4.95\%$	78	539 ± 112
ConvLSTM (center)	$74.99\% \pm 7.42\%$	$74.40\% \pm 9.76\%$	78	539 ± 112
ConvLSTM (hip)	$71.74\% \pm 4.51\%$	$71.12\% \pm 7.81\%$	78	539 ± 112
MS-G3D (baseline)	$74.24\% \pm 6.39\%$	$74.15\% \pm 7.20\%$	79	835 ± 55
MS-G3D (center)	$68.71\% \pm 6.58\%$	$69.69\% \pm 7.24\%$	79	835 ± 55
MS-G3D (hip)	$68.90\% \pm 5.12\%$	$70.47\% \pm 4.76\%$	79	835 ± 55
ST-GCN (baseline)	$75.21\% \pm 7.26\%$	$75.03\% \pm 7.54\%$	82	612 ± 85
ST-GCN (center)	$76.95\% \pm 6.62\%$	**$76.96\% \pm 7.02\%$**	82	612 ± 85
ST-GCN (hip)	$71.87\% \pm 5.12\%$	$74.64\% \pm 6.16\%$	82	612 ± 85
TinierHAR (baseline)	$62.39\% \pm 6.92\%$	$58.67\% \pm 9.87\%$	81	538 ± 84
TinierHAR (center)	$70.58\% \pm 13.43\%$	$72.17\% \pm 13.25\%$	81	538 ± 84
TinierHAR (hip)	$73.37\% \pm 9.64\%$	$73.60\% \pm 9.95\%$	81	538 ± 84
TinyHAR (baseline)	$71.30\% \pm 6.87\%$	$66.48\% \pm 9.12\%$	79	534 ± 51
TinyHAR (center)	**$76.97\% \pm 3.83\%$**	$75.94\% \pm 5.85\%$	79	534 ± 51
TinyHAR (hip)	$70.28\% \pm 5.48\%$	$69.44\% \pm 7.79\%$	79	534 ± 51

TABLE VII: Results for sliding window configuration (150/75)

Approach	Accuracy	macro F_1-Score	GPU [Wh]	VRAM [MB]
ConvLSTM (baseline)	$53.63\% \pm 9.51\%$	$52.23\% \pm 8.46\%$	84	540 ± 104
ConvLSTM (center)	$75.09\% \pm 3.40\%$	$75.03\% \pm 5.98\%$	84	540 ± 104
ConvLSTM (hip)	$73.29\% \pm 8.67\%$	$73.22\% \pm 9.89\%$	84	540 ± 104
MS-G3D (baseline)	$76.79\% \pm 7.52\%$	$76.08\% \pm 8.58\%$	81	901 ± 72
MS-G3D (center)	$72.74\% \pm 10.04\%$	$73.29\% \pm 10.72\%$	81	901 ± 72
MS-G3D (hip)	$76.86\% \pm 7.36\%$	$77.16\% \pm 7.72\%$	81	901 ± 72
ST-GCN (baseline)	$76.54\% \pm 4.29\%$	$75.64\% \pm 3.49\%$	87	660 ± 85
ST-GCN (center)	**$78.80\% \pm 7.48\%$**	**$78.52\% \pm 8.20\%$**	87	660 ± 85
ST-GCN (hip)	$77.25\% \pm 8.37\%$	$78.24\% \pm 9.51\%$	87	660 ± 85
TinierHAR (baseline)	$71.44\% \pm 8.61\%$	$68.16\% \pm 7.05\%$	75	524 ± 120
TinierHAR (center)	$75.49\% \pm 5.17\%$	$75.41\% \pm 8.24\%$	75	524 ± 120
TinierHAR (hip)	$72.53\% \pm 7.90\%$	$72.13\% \pm 11.28\%$	75	524 ± 120
TinyHAR (baseline)	$72.36\% \pm 6.42\%$	$68.43\% \pm 6.70\%$	77	535 ± 44
TinyHAR (center)	$78.55\% \pm 3.68\%$	$78.47\% \pm 5.60\%$	77	535 ± 44
TinyHAR (hip)	$72.39\% \pm 5.72\%$	$72.43\% \pm 8.64\%$	77	535 ± 44

TABLE VIII: Results for sliding window configuration (180/90)

Approach	Accuracy	macro F_1-Score	GPU [Wh]	VRAM [MB]
ConvLSTM (baseline)	$47.26\% \pm 14.66\%$	$46.24\% \pm 19.86\%$	78	531 ± 126
ConvLSTM (center)	$78.27\% \pm 7.24\%$	$78.21\% \pm 8.52\%$	78	531 ± 126
ConvLSTM (hip)	$74.28\% \pm 4.73\%$	$73.38\% \pm 7.29\%$	78	531 ± 126
MS-G3D (baseline)	$77.18\% \pm 7.70\%$	$75.76\% \pm 8.87\%$	82	995 ± 81
MS-G3D (center)	$76.54\% \pm 8.26\%$	$78.72\% \pm 6.98\%$	82	995 ± 81
MS-G3D (hip)	$71.66\% \pm 6.74\%$	$73.23\% \pm 6.43\%$	82	995 ± 81
ST-GCN (baseline)	$77.13\% \pm 9.16\%$	$76.44\% \pm 9.98\%$	83	681 ± 96
ST-GCN (center)	**$79.65\% \pm 7.70\%$**	**$79.86\% \pm 7.03\%$**	83	681 ± 96
ST-GCN (hip)	$79.38\% \pm 6.48\%$	$78.88\% \pm 8.84\%$	83	681 ± 96
TinierHAR (baseline)	$66.90\% \pm 6.43\%$	$62.40\% \pm 7.35\%$	76	535 ± 98
TinierHAR (center)	$74.10\% \pm 3.73\%$	$74.13\% \pm 5.47\%$	76	535 ± 98
TinierHAR (hip)	$74.74\% \pm 6.28\%$	$76.15\% \pm 6.60\%$	76	535 ± 98
TinyHAR (baseline)	$72.38\% \pm 6.01\%$	$68.91\% \pm 9.14\%$	76	536 ± 49
TinyHAR (center)	$78.03\% \pm 6.20\%$	$78.22\% \pm 5.59\%$	76	536 ± 49
TinyHAR (hip)	$70.63\% \pm 8.04\%$	$69.64\% \pm 11.02\%$	76	536 ± 49