Multivariate Human Activity Segmentation: Systematic Benchmark with ClaSP

Arik Ermshaus[1][0000-0002-8138-3060], Patrick Schäfer[1], and Ulf Leser[1]

Humboldt University of Berlin, Berlin, Germany
{ermshaua,patrick.schaefer,leser}@informatik.hu-berlin.de

Abstract. Human activity recognition (HAR) systems extract activities from observational data, such as sensor measurements from mobile devices, to provide for instance medical, fitness, or security information. A crucial initial step in these data analysis workflows is segmenting continuous numerical measurements into variable-sized segments that correspond to single activities. Human activity segmentation (HAS) enables downstream classification algorithms to label entire activities. Unfortunately, current time series segmentation (TSS) algorithms exhibit limited performance on multivariate sensor data due to complex temporal dynamics and irrelevant dimensions. This limits their applicability in HAR workflows. In this review, we provide a systematic benchmark of dimensionality reduction, model aggregation and change point selection applied to the ClaSP TSS algorithm for real-world, multidimensional mobile sensing data. We evaluated the accuracy of the techniques in an experimental study using 250 data sets from the HAS challenge at ECML/PKDD and AALTD 2023. Our findings indicate that extending ClaSP for multivariate data, by aggregating internal representations, yields better results compared to reducing data dimensionality or selecting change points (CPs) from different channels. We report a new state of the art with 73% average accuracy on the challenge benchmark.

Keywords: Ubiquitous Sensing · Human Activity Recognition · Data Mining · Unsupervised Learning · Time Series Segmentation

1 Introduction

Monitoring human behaviour can provide crucial insights into personal health, fitness levels, and tactical arrangements of activities [1]. The sequence, duration, and classification of these activities are especially important as they are the basis for human processes. Health professionals, for instance, examine these processes to detect and monitor medical conditions [2], while military personnel analyse movement sequences to enhance decision-support systems [1]. To derive processes from monitoring human behaviour, data is acquired through videos, environmental sensing, or wearable devices [3]. Smart devices are particularly valuable as they are worn consistently and can capture human behaviour using inertial measurement units (IMUs). These sensors typically include accelerometers, gyroscopes, and magnetometers, each producing triaxial measurements

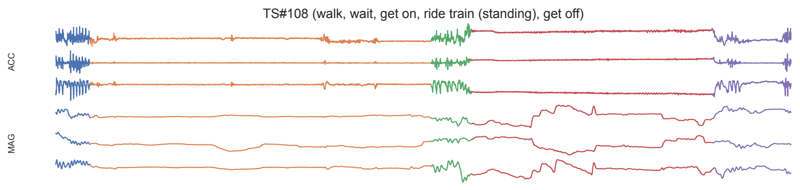


Fig. 1. MTS from the HAS challenge benchmark [4]. The different channels show sensor signals capturing a student commuting to university, with activities colour-coded. Only a subset of dimensions is relevant for HAS.

sampled at hundreds of Hertz (Hz). The resulting recordings form a multivariate time series (MTS), containing vectors of sensor measurements.

Figure 1 illustrates an example of a student commuting to university by train. The MTS captures human activities as extended periods of similar temporal patterns, such as waiting (orange) versus boarding (green). These patterns are repeated within processes and vary in shape and statistical properties across different activities. Note, acceleration (top) and magnetometer (bottom) readings are not consistent, which complicates knowledge discovery from such MTS.

To learn processes from human mobile sensing data, activities must be located and labelled. The extensive literature on human activity recognition (HAR) offers a comprehensive toolbox for feature extraction, classification, and software for this task [3]. One of the initial preprocessing steps in HAR involves segmenting MTS into smaller, consecutive parts, which are then classified using machine learning (ML) technology. This task can be approached by dividing the MTS into equal-sized subsequences or by segmenting it into variable-sized partitions, each corresponding to a single activity. The latter approach, known as human activity segmentation (HAS), is advantageous because it learns the start times and durations of activities, providing downstream tasks with clearly defined data segments corresponding to single activity labels [5]. HAS is generally referred to as time series segmentation (TSS), which divides a time series (TS) into homogeneous segments, separated by abrupt transitions called change points (CPs) [6].

Univariate TSS (UTSS) has been extensively studied and benchmarked [7, 8], while multivariate TSS (MTSS) has received less attention. This is particularly problematic for activity segmentation, which requires multivariate signals to detect complex behaviours, such as dance or fight moves [9]. To address this shortcoming, the 8th Workshop on Advanced Analytics and Learning on Temporal Data (AALTD@ECML)[1] conducted the ECML/PKDD 2023 HAS discovery challenge [4]. The competition, featuring 57 participants, aimed to improve the performance of multidimensional human activity segmentation using a new benchmark data set of 250 MTS capturing 100 activities performed by 15 students in 6 motion sequences. Both winners of the challenge utilized adaptations of the ClaSP method to apply it to multivariate measurements [10, 11]. ClaSP has demonstrated superior performance for UTSS across multiple bench-

[1] https://ecml-aaltd.github.io/aaltd2023

marks for both batch [8] and streaming data [12]. To further explore its potential for offline MTSS, this benchmark reviews, categorizes, and evaluates currently available and new implementations of multivariate ClaSP to identify the most promising variant for multivariate HAS. Yet, not all studied methods are limited to ClaSP. Specifically, this paper's contributions are:

1. We review three different categories of approaches from the literature to modify the ClaSP method for offline MTSS, namely: dimensionality reduction, model aggregation and CP selection / ensembling. This includes technical descriptions, computational complexity analyses, and examples of HAR.
2. We conducted ablation studies for the three strategies and benchmarked their performances across the 250 HAS challenge data sets. Our findings show that model aggregation using distance averaging achieves the highest accuracy of 73%, setting a new state of the art.
3. To promote further development of MTSS algorithms, we provide all source codes, data sets, and extended evaluations on our supporting website [13].

The remainder of this paper contains background definitions (Section 2), related work (Section 3), the review and empirical evaluation of the three categories (Section 4 and 5), as well as a conclusion (Section 6).

2 Definitions and Background

We define the formal concepts of time series, subsequences, time series segmentation, and the ClaSP algorithm that we use throughout this paper.

Definition 1. *A time series (TS) T is an ordered sequence of $n \in \mathbb{N}$ real vectors $T = (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_n) \in \mathbb{R}^{n \times d}$ that constitutes the measurements from $d \in \mathbb{N}$ sensors of an activity routine of length n. The j-th sensor data is in $T^{(j)} \in \mathbb{R}^n$.*

If T has $d = 1$ dimension (aka channel), it is called a univariate TS (UTS); otherwise, it is a multivariate TS (MTS). The measurements within T are recorded at equidistant intervals, e.g., one data vector $\boldsymbol{t}_i$ every 20 milliseconds.

In this way, we study MTS from IMU units in smartphones, which contain accelerometers, gyroscopes, and magnetometers [3]. Accelerometers capture the acceleration forces on a mobile phone, indicating motion presence. Gyroscopes measure angular velocity, inferring rotation during activities. Magnetometers record the geomagnetic field's impact on the smartphone, providing orientation information. IMU sensors capture measurements from the X, Y, and Z axes and are sampled at a few hundred Hertz (Hz), leading to long MTS that contain activities as recurring patterns. See Figure 1 for an example.

Definition 2. *Given a TS T, a* subsequence *$T_{s,e}$ of T, with start and end offset s and e, is the d-dimensional window of contiguous measurements from T at position s to position e, i.e., $T_{s,e} = (\boldsymbol{t}_s, \ldots, \boldsymbol{t}_e)$, with $\boldsymbol{t}_i \in \mathbb{R}^d$ and $1 \leq s \leq e \leq n$. The width of $T_{s,e}$ is $|T_{s,e}| = e - s + 1$.*

Subsequences that capture single parts of an activity (e.g., taking a step) are called temporal patterns, and their length is the window size. TS with human behaviour contain parts with periodic subsequences constituting a longer activity (such as walking), which may suddenly change or gradually drift into another one (e.g., running). This manifests as changes in width, amplitude, or shape [14].

Definition 3. *For a TS T, capturing a motion routine, a* change point *(CP) is an offset $i \in [1, \dots, n]$ corresponding to an abrupt transition between two activities. A* segmentation *of T is the ordered sequence of CPs in T, i.e., $t_{i_1}, \dots, t_{i_S}$ with $1 < i_1 < \cdots < i_S < n$ at which the captured behaviour changed activities.*

The task of human activity segmentation (HAS) is to find the segmentation of a TS that corresponds to the sequence of captured activities. Specifically, an algorithm must find all CPs that divide the TS into segments. This is an unsupervised machine learning (ML) problem generally referred to as time series segmentation (TSS) or change point detection (CPD) [6]. For MTS, it requires procedures to detect long stretches of homogeneous segments by selecting relevant channels and comparing subsequences by shape or statistical properties. A recent univariate TSS technique, which we study, is the ClaSP algorithm [8].

Definition 4. *Given an UTS T and a subsequence width w, ClaSP is a real-valued sequence of length $n - w + 1$, in which the i-th value marks the cross-validation score $c_i \in [0, 1]$ of a classifier trained on a binary classification problem with overlapping labelled subsequences $[(T_{1,w}, \boldsymbol{0}), \dots, (T_{i-w+1,i}, \boldsymbol{0}), (T_{i-w+2,i+1}, \boldsymbol{1}), \dots (T_{n-w+1,n}, \boldsymbol{1})]$, with labels 0 and 1.*

The central idea of ClaSP is to frame TSS as a collection of self-supervised, binary subsequence classification problems. Each cross-validation score c_i reports how well a TS classifier can differentiate the left from the right subsequences (see Figure 5 for an example). All scores constitute a profile, that annotates T, show pronounced peaks for CPs and valleys during segments. This course is used for detecting CPs by finding the peaks [8].

The ClaSP concept has been efficiently implemented for batch [8] and streaming [12] settings using a k-nearest neighbour (k-NN) classifier. However, ClaSP originally has only been developed for UTS. To apply the idea also to MTS, several papers have proposed offline ensembling strategies to aggregate ClaSP profiles or the extracted CPs from multiple channels [15, 10, 11]. In this work, we review and systematically benchmark these proposals and new variants.

3 Related Work

In recent years, there has been much progress in human activity recognition (HAR) systems, devices, and experimental studies [1]. This research field is versatile, encompassing various behaviours, data acquisition techniques, and analysis methods [3]. Some studies focus on the streaming setting, where activities are interpreted in real-time, providing users with immediate feedback, e.g. on

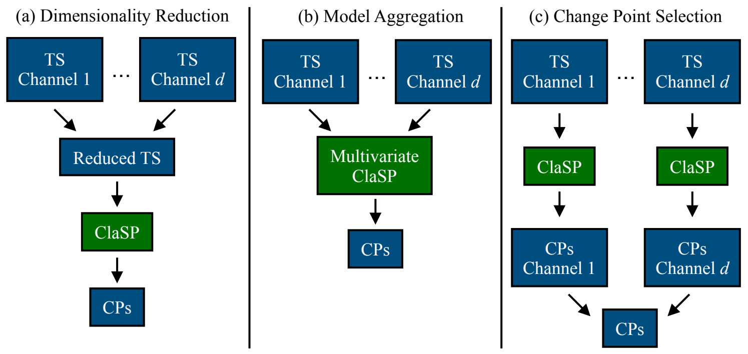


Fig. 2. Data flow for MTSS approaches. Blue boxes show input and output data. Green boxes display ClaSP component.

their fitness status. Others analyse human behaviour post-hoc, deriving insights such as those from gait analysis [2].

Human activity segmentation (HAS) is a preprocessing step in HAR workflows. It partitions sensor data into variable-sized windows, each containing a single activity, which is then further processed [3]. The difficulty of this task is to select relevant sensor data for segmentation, ignore noise and misleading observations. Multivariate time series segmentation (MTSS) methods achieve this by dividing multidimensional numerical measurements into homogeneous slices based on shape or statistical properties [6]. A common approach is to frame this task as an optimization problem using a user-selected cost function, such as L2 or autoregressive cost, and applying a change point (CP) constraint. This problem can be solved with an exact algorithm like PELT or an approximation like BinSeg [6]. Specifically for multivariate sensor data from human activities, Sadri et al. proposed a cost function based on information gain [16]. ESPRESSO is another technique for multidimensional HAS that extracts CPs from a weighted subsequence arc curve and uses entropy to validate them [17]. The arc curve, proposed originally by Gharghabi et al., measures the density of similar subsequences in potential segments [7].

In this study, we explore the ClaSP algorithm in an offline multivariate setting for HAS. Initially proposed for UTS [8], ClaSP has demonstrated superior results on multiple HAS benchmarks compared to the aforementioned approaches [5, 4]. We review and evaluate several extensions of multivariate ClaSP to find a suitable implementation for HAR.

4 Multivariate Time Series Segmentation with ClaSP

In this section, we review three method-agnostic approaches for offline MTSS, we implemented with ClaSP: (a) dimensionality reduction, (b) model aggregation, and (c) CP selection / ensembling. The approaches aggregate multivariate into one-dimensional data at different stages of the segmentation process. Dimensionality reduction directly projects the multivariate input TS to a univariate one. An UTSS algorithm then processes this synthesized series to extract CPs. Model aggregation extends internal data structures for multivariate input, merging them into univariate aggregates for segmentation. CP selection / ensembling computes segmentations separately for each TS dimension, then filters out or merges (near-)duplicate CPs.

Figure 2 shows the data flow of the approaches. Dimensionality reduction and CP selection / ensembling operate directly on the input or output data (blue boxes) and are method-independent. Model aggregation modifies ClaSP's internal components (green box) and requires changes to the original algorithm.

In the following subsections 4.1 to 4.3, we review the three strategies (a to c) in detail. We provide own pseudocodes, analyse their implementations and computational complexities, as well as discuss advantages and shortcomings.

4.1 Dimensionality Reduction

Univariate TS analytics can be applied to MTS by reducing their dimensionality. Tanaka et al. [18] explored this approach in the context of motif discovery. The main idea is to treat MTS channels as features and time points as instances. Then, a dimensionality reduction technique aggregates these features into a single component. Dimensionality reduction methods project multidimensional features of a data set into a lower-dimensional target space while preserving important relationships between samples as good as possible, such as distributions, separability, or distances [19]. This is achieved by merging correlated features, identifying independent components, or projecting features. Popular techniques include principal component analysis (PCA), independent component analysis (ICA), random projection (RP), and autoencoders.

Algorithm 1 implements this idea. It takes a MTS T, $|T| = n$ with d dimensions as input and applies a dimensionality reduction technique, such as PCA [19], to reduce the d channels into one. UTSS algorithms, such as ClaSP, can then process this synthesized TS. Reducing TS dimensionality decreases noise and redundant information from data. However, it can also over-simplify complex temporal dynamics or destroy inter-channel dependencies.

The computational complexity of Algorithm 1 mainly depends on the dimensionality reduction method used [19]. For instance, PCA requires $\mathcal{O}(n{\cdot}d^2{+}d^3)$ for calculating the covariance matrix and singular value decomposition (SVD). ICA needs $\mathcal{O}(n{\cdot}d^2{\cdot}i)$ for i iterations, and random projection takes $\mathcal{O}(n{\cdot}d{\cdot}l) = \mathcal{O}(n{\cdot}d)$ for multiplying X with a random projection matrix of size $(d \times l)$, where $l = 1$ is the size of the lower-dimensional target space. Regarding space complexity, PCA and ICA each require $\mathcal{O}(n \cdot d + d^2)$ to store X and the covariance (or unmixing) matrix. Random projection requires $\mathcal{O}(n \cdot d + d \cdot l)$, which reduces to $\mathcal{O}(n \cdot d)$.

Figure 3 illustrates an example of dimensionality reduction. It shows an indoor sports routine performed by a 25-year-old male, captured by three triaxial inertial measurement unit (IMU) smartphone sensors as a MTS (top-9 channels). Activities are coloured differently. The bottom-3 UTS display the synthesized TS

Algorithm 1 Dimensionality Reduction

```
1: procedure ReduceDimensions($T$)
2:     $X \leftarrow$ create data set matrix from $T$ of size $(n \times d)$
3:     $R \leftarrow$ PCA($n\_components = 1$).fit_transform($X$)  ▷ Apply reduction technique.
4:     return $R$
5: end procedure
```

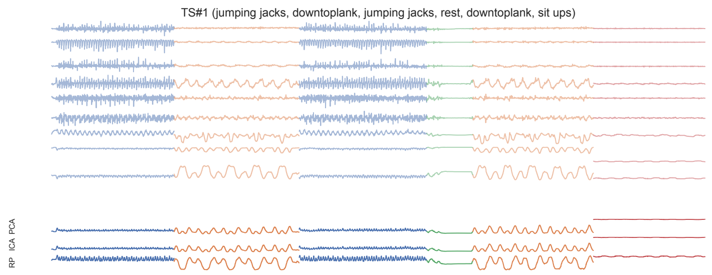


Fig. 3. TS dimensionality reduction with PCA, ICA and RP. The top-9 channels show sensor signals capturing indoor sport activities [4] (different light colours). The bottom-3 TS illustrate the results of PCA, ICA and RP (full colours).

using PCA, ICA, and RP. Each technique uniquely reduces the MTS but retains similarities within activities (e.g., two instances of "down to plank") and separability across activities (e.g., "jumping jacks" versus "sit-ups"). This indicates that dimensionality reduction can be a viable preprocessing for MTSS.

4.2 Model Aggregation

Instead of directly reducing a MTS to an UTS, model aggregation adapts the TSS procedure for multivariate data. The central idea is to compute the TSS model per channel and aggregate the resulting features, allowing for potentially more meaningful representations compared to raw measurements. Model aggregation is well suited for the ClaSP algorithm that involves a two-step feature transformation, k-NN model, and cross-validation score profile, both of which can be extended for MTS and aggregated.

Distance Averaging: To create the k-NN model for MTS, we calculate distances between subsequences for each channel separately using STOMP, which employs optimization techniques to speed-up calculations [20]. Then, we average these distances and thereafter select the k smallest values, applying an exclusion radius to ignore self-matches. For constructing the k-NN in ClaSP, we calculate the z-normalized Euclidean distances between subsequences of width w.

Algorithm 2 outlines this process. It takes a MTS T, $|T| = n$ with d channels, a subsequence width w, and a position p as input and averages the pairwise distances between $T^{(i)}_{p,p+w-1}$ and all other subsequences in $T^{(i)}$ for each of the d dimensions. This includes proximity information from each dimension, weighing

Algorithm 2 Distance Averaging

```
1: procedure AVERAGEDISTANCES($T$, $w$, $p$)
2:     $D_{global} \leftarrow$ array of length $|T| - w + 1$ with 0s        ▷ Initialize distances.
3:     for $i \in [1, \dots, d]$ do        ▷ Update distances for each dimension.
4:         $D_{global} \leftarrow D_{global} + \text{CALC\_DISTANCES}(T^{(i)}, w, p))$
5:     end for
6:     return $\frac{1}{d} \cdot D_{global}$        ▷ Return average distances.
7: end procedure
```

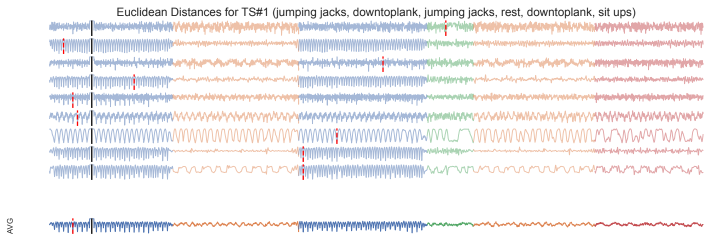


Fig. 4. Euclidean distances between the subsequence at position 500 (black bar) and all other subsequences (width 50) for the TS from Figure 3 (light colours). 1-NNs are illustrated as red bars. Ideally, they should be located in one of the blue segments. The bottom distances are the averages (full colours).

them equally. It captures complex multivariate dynamics, but is sensitive to noise or irrelevant channels. Thereafter, the k smallest values are determined.

The runtime complexity of Algorithm 2 is mainly driven by the distance calculation, which is $\mathcal{O}(n)$ for a single channel using STOMP [20]. As it is executed d times, the overall runtime is $\mathcal{O}(n \cdot d)$, increasing the complexity of ClaSP from $\mathcal{O}(n^2)$ for the univariate case to $\mathcal{O}(d \cdot n^2)$ for MTS with d dimensions. The space complexity for distance averaging, and for ClaSP, is $\mathcal{O}(d \cdot n)$.

Figure 4 demonstrates Algorithm 2 for the TS from Figure 3 and the subsequence at position 500 (black bar, width 50). The top-9 series show the distances for each channel, and the bottom series shows the averaged distances, the output of the procedure. The red bars indicate the 1-NN subsequence per channel. The first channel wrongly assigns a subsequence from the resting activity, which could lead to an inaccurate segmentation. All other dimensions correctly find a subsequence from the first and second instance of jumping jacks. The averaged distances also identify the 1-NN belonging to the first segment.

Distance-based Channel Selection: Distance averaging assumes all TS channels to be equally important, which may not be true in practice. Often, only a fraction of the dimensions contain relevant information for segmentation (e.g., top 2–9 distances in Figure 4). Considering all channels in such scenarios introduces unnecessary noise into the distance calculations. To address this, we can modify Algorithm 2 to select distances based on specific criteria.

One simple approach is to consider only the f out of d distance vectors with increasing global minima. This subgroup has the smallest average 1-NN

Algorithm 3 Profile Averaging

```
1: procedure AVERAGEPROFILES($T$, $w$)
2:     $P_{global} \leftarrow$ array of length $|T| - w + 1$ with 0s        ▷ Initialize profile.
3:     for $i \in [1, \ldots, d]$ do        ▷ Update scores for each dimension.
4:         $P_{global} \leftarrow P_{global} + \text{CALC\_CLASP}(T^{(i)}, w)$
5:     end for
6:     return $\frac{1}{d} \cdot P_{global}$        ▷ Return average profile.
7: end procedure
```

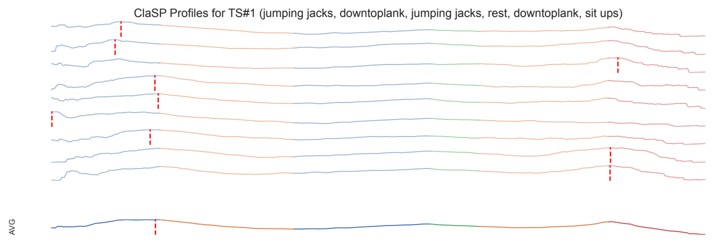


Fig. 5. ClaSP profiles for the TS from Figure 3 (light colours). Global maxima (CPs) are illustrated as red bars. Ideally, they should capture one of the activity transitions. The bottom profile is the average (full colours).

distance. The parameter f can be user-defined or learned through thresholding. A more sophisticated version of this technique is implemented in the mSTAMP algorithm [21] for motif discovery. It calculates distances for all dimensions, sorts them in ascending order per time point, and averages the first f out of d distance vectors, fitting best for segmentation. This subgroup not only includes the smallest 1-NN distances, but also the smallest overall distances at each offset. Yet, the f dimensions are not fixed, but can change.

Profile Averaging: Score profiles annotate the likelihood of CPs for each time point. They can be applied to each channel of a MTS, with the resulting profiles being averaged before applying segmentation. This concept was discussed by Wang et al. [15] for ClaSP and is also implemented for FLUSS [7].

Algorithm 3 implements this idea. It takes a MTS T, $|T| = n$ with d channels and a subsequence width w as input, computes ClaSP for each of the d channels and averages the resulting profiles, which is used for segmentation. Similar to distance averaging, it considers each profile as an equal contribution. However, it assumes that individual channels contain sufficient differences to identify segment transitions, recognized by the k-NN classifier and represented as peaks in the profiles. In contrast, distance averaging detects differences spread across multiple dimensions, recognized only after combining distances.

The computational complexity of Algorithm 3 depends on the ClaSP routine, which runs in $\mathcal{O}(n^2)$ per UTS. This process repeats for each channel, resulting in a total runtime complexity of $\mathcal{O}(d \cdot n^2)$. The space complexity for a single ClaSP is $\mathcal{O}(n)$, which extends linearly to $\mathcal{O}(d \cdot n)$ in the multivariate setting.

Algorithm 4 Change Point Selection / Ensembling

```
1: procedure FilterChangePoints($T$)
2:     $C_{all} \leftarrow$ empty set                              ▷ Initialize CP set.
3:     for $i \in [1, \ldots, d]$ do                          ▷ Add CPs for each dimension.
4:         $C_{all} \leftarrow C_{all} \cup$ CLASP_SEGMENTATION($T^{(i)}$)
5:     end for
6:     return FILTER($C_{all}$)                ▷ Filter / merge (near-)duplicate CPs.
7: end procedure
```

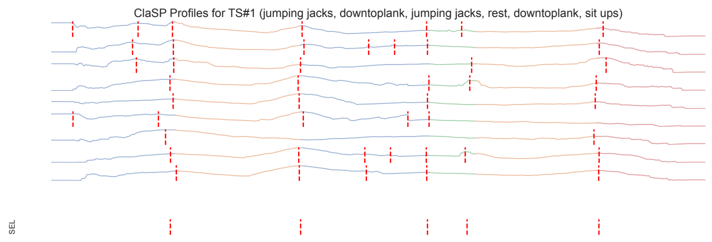


Fig. 6. ClaSP profiles for the TS from Figure 3 (light colours). All found CPs are illustrated as red bars. Single channels contain false or missing predictions. The bottom CPs are the filtered selection and correctly capture the activity transitions.

Figure 5 illustrates how Algorithm 3 computes ClaSP per channel (top-9 profiles) and then averages them (bottom profile). The red bars indicate the highest-scoring CPs per channel (global maxima), mainly located at the first transition (jumping jacks to plank) and the last change (plank to sit-ups). However, the CPs from the first two (and fifth) dimensions substantially deviate from the ground truth. The average profile correctly aggregates the CP information, identifying the first transition as the most substantial CP.

Score-based Channel Selection: Similar to distance averaging, Algorithm 3 assigns equal weight to each channel during aggregation, which may wrongly include noisy or misleading scores (see profiles 1 & 2 in Figure 5). To address this, we can apply different selection strategies to filter dimensions.

ClaSP uses significance testing to validate CPs in segmentation. We can apply this testing before averaging to include only profiles with significant CPs. However, this may exclude channels showing trends in their ClaSP profiles that might become significant when combined. Another selection technique, used in ClaSP ensembling, is to maximize scores rather than averaging them. This approach selects the best-performing ClaSP by design and disregards all but one channel in MTS aggregation, potentially filtering out useful channels. We can counteract this by selecting the top f out of d best-scoring profiles.

4.3 Change Point Selection / Ensembling

Instead of modifying a MTS or the segmentation model, we can perform TSS on each channel separately and only merge the resulting CPs. This technique is invariant to the segmentation procedure and primarily relies on CP filtering, akin to dimensionality reduction. The authors of [11] propose an interval weighting scheme, while Harańczyk uses threshold-based pruning [10]. Another approach is to use agglomerative CP clustering.

Algorithm 4 details CP selection. It takes a MTS T, $|T| = n$ with d channels as input, computes the CPs for all d dimensions separately and filters them, e.g., using threshold-based pruning [10], to remove near-duplicates and return unique CPs. Similar to profile averaging, this procedure can only identify CPs recognized per channel, not those requiring the combination of dimensions.

The runtime complexity of Algorithm 4 primarily depends on ClaSP and the specific filtering mechanism. Running the univariate ClaSP segmentation per channel costs $\mathcal{O}(C \cdot n^2)$ for C detected CPs. Extending this to the multivariate setting results in $\mathcal{O}(d \cdot C_{max} \cdot n^2)$ for a maximum of C_{max} detected CPs per channel. Applying the filtering mechanism adds $\mathcal{O}(d \cdot C_{max})$ for interval weighting, $\mathcal{O}(C_{all}^2)$ for threshold-based pruning, and $\mathcal{O}(C_{all}^3)$ for agglomerative clustering. The space complexity of the entire Algorithm 4 is $\mathcal{O}(d \cdot n)$, except for agglomerative clustering, which adds $\mathcal{O}(C_{all}^2)$ for storing the CP distance matrix.

Figure 6 shows Algorithm 4 for the MTS from Figure 3. The top-9 series are ClaSP profiles with all detected CPs per channel, and the bottom CPs are filtered using agglomerative clustering, reporting the mean CP per cluster. While individual channels produce comparable but heterogeneous segmentations with a mix of true, false, and missing CP predictions, the filtered output correctly captures the true activity transitions. Thus, it can be favourable to ensemble CPs from multiple channels, compared to selecting predictions from one dimension.

5 Experimental Evaluation

We empirically investigate the accuracy of the three strategies for MTSS using human activity data. First, we describe the used mobile sensing data sets and evaluation measure in Subsection 5.1. We then explore design choices per approach in an ablation study (see Subsection 5.2). Lastly, we compare the performance of the techniques in Subsection 5.3 with 3 univariate baselines and explore the segmentation of an use case. All experiments were conducted on an Intel Xeon 8358 with 2.60 GHz, 2 TB RAM, 128 cores, running Python 3.8. To foster reproducibility and replicability of our results, we provide source codes, data sets and raw measurements on our supporting website [13].

5.1 Setup

HAS Challenge Data Sets: We utilize the 250 MTS from the Human Activity Segmentation Challenge [4], held at ECML/PKDD and the AALTD 2023 workshop. This multi-modal benchmark originates from 40 twelve-dimensional smartphone recordings, capturing 15 students performing 6 distinct motion sequences in both indoor and outdoor settings. Each data set includes values from 6-9 out of 12 sensors: triaxial accelerometer, gyroscope, magnetometer as well as latitude, longitude, and speed. The MTS are synchronized and sampled at 50 Hz, annotated with activity labels and transition offsets as ground truth. They range from 7 seconds to 14 minutes in duration (median: 100 seconds) and contain between 1 and 15 segments (76% of MTS have 5 or fewer). The duration of single activities varies from half a second (waiting) to 10 minutes (running). The 250 MTS come in fixed randomly split public and private sets (125 MTS each) to avoid overfitting; the public set is used for design choices, and the private set is used for competitor comparisons. For an example TS, see Figure 1 or 3 again.

Covering Score: We use the Covering evaluation measure [5], to assess the performance of MTSS methods on a given HAS data set. It measures how well the predicted segments overlap with the annotated ones using the Jaccard index.

Specifically, let T, $|T| = n$ be a MTS with ground truth CPs $cpts_{true}$ and predicted CPs $cpts_{pred}$, each located in $[1, \dots, n]$. We consider the interval of successive CPs $[t_{i_k}, \dots, t_{i_{k+1}}]$ a segment in T, and let $segs_{true}$ and $segs_{pred}$ be the sets of ground truth and predicted segmentations, respectively. We define $t_{i_0} = 0$ as the first and $t_{i_C} = n + 1$ as the last CP to include the first (last) segment. Covering reports the best-scoring weighted overlap between the true and predicted segmentations as a value in the interval $[0, \dots, 1]$ (higher is better):

$$Covering = \frac{1}{\|T\|} \sum_{s \in segs_{true}} \|s\| \cdot \max_{s' \in segs_{pred}} \frac{\|s \cap s'\|}{\|s \cup s'\|} \quad (1)$$

This allows for the comparison of sets $cpts_{true}$ and $cpts_{pred}$ with varying lengths, including empty sets. To compare different methods across multiple data sets, we aggregate the Covering scores into a single ranking. We compute the rank of each technique per data set, i.e., the best method is assigned rank 1, the second-best rank 2, etc., and average the ranks over all data sets per method to obtain its overall rank. For illustration, we use critical difference (CD) diagrams [22]. The best approaches, with the lowest average ranks, are shown to the right of the diagram (see Figure 7). Groups of methods with insignificantly different performances are connected by a bar, based on a pairwise one-sided Wilcoxon signed-rank test with $\alpha = 0.05$ and Holm correction.

Hyperparameters: We set different hyperparameters for the approaches and fix them for all evaluations. The dimensionality reduction techniques reduce a MTS to a lower-dimensional target space. We use the first component for all tested methods and draw random values from a normal distribution for random projection. ClaSP processes the synthesized TS with default parameters.

The selection strategies for model aggregation require the number of selected dimensions, which we set to $\lfloor \frac{d}{2} \rfloor$ for all techniques that require this parameter. As the choice of this parameter is generally use case dependent, we study the overall impact of excluding dimensions in the segmentation. The multivariate ClaSP, used for model aggregation, first learns a window size per channel and then uses the minimal value for remaining computations. It also sets a stricter significance level for CP validation of $1e-30$ (compared to univariate ClaSP's default of $1e-15$), which overall leads to more accurate results.

For CP ensembling, ClaSP processes each channel with default parameters. Agglomerative clustering uses average linkage thresholded at $5 \cdot w$. Clusters of CPs at or above this distance are not merged, following ClaSP's strategy that segments must have at least $5 \cdot w$ data points. Pruning and weighting-based CP detection make more custom design choices, as outlined in the respective publications [10, 11]. We evaluate the winning challenge submissions.

Additionally, we report the scores for the univariate BinSeg, ClaSP, and FLUSS challenge baselines, which only segment Y-axis acceleration [4].

5.2 Ablation Study

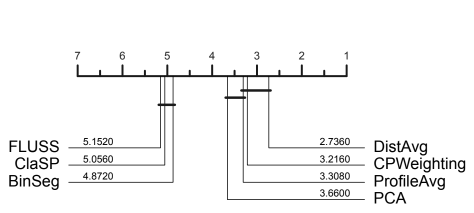
We tested different design choices for each of the three approaches of MTSS with ClaSP on the public HAS challenge data split (125 TS). We analyse the performances per approach to identify variants for comparative evaluation. CD diagrams and more detailed analyses are on our supporting website [13].

Dimensionality Reduction: We tested PCA, ICA, and RP for dimensionality reduction. PCA (1.93) ranked first, followed by RP (1.96) and ICA (2.11). The differences in rank are not statistically significant. Considering average Covering, RP ($67 \pm 24\%$) leads, followed by PCA ($64 \pm 26\%$) and ICA ($63 \pm 27\%$). We find that the specific implementation only leads to differences in tendencies. Both RP and PCA are good candidates for this approach. We choose PCA for further analysis as it scores the lowest average rank.

Model Aggregation: We evaluated distance averaging with all dimensions, using the f out of d distances with the smallest minima, and dimension sorting for selection. Using all channels ranks first (1.93), followed by using f out of d distances (1.99), and dimension sorting (2.08). Differences in rank are not significant. However, using all distances also has the highest Covering of $74 \pm 21\%$ on average and 78% in median, compared to the selection strategies. We conclude that distance-based channel selection does not have an advantage for the TS.

For profile averaging, we tested using all dimensions, aggregating f out of d profiles with the highest maxima, and averaging profiles with significant CPs. Using f out of d profiles ranks first (1.88), averaging all profiles ranks second (1.9), and aggregating profiles with significant CPs ranks last (2.22). While the two best-scoring variants do not show significant differences in rank, both are significantly better than the last. Summary statistics confirm the ranking (see [13]). Similar to distance averaging, the selection strategies do not show any substantial improvement in scores or ranks. On the contrary, averaging profiles with significant CPs even leads to significantly worse results.

We find that for both distance and profile averaging, it could be the case that (a) most dimensions positively contribute to the segmentation, (b) the selection criteria do not work as well, or (c) the hyper-parameter setting the number of selected dimensions needs more careful adjustment. Based on our results, we use all channels in model aggregation for further comparison.



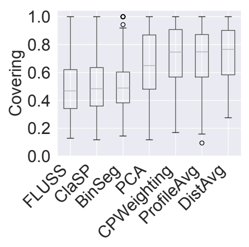


Fig. 7. CD diagram (left) and box plot (right) on the private HAS challenge split for the univariate challenge baselines (BinSeg, ClaSP, FLUSS) as well as dimensionality reduction with PCA, distance / profile averaging, and CP weighting.

CP Selection / Ensembling: We assessed CP selection and ensembling using pruning, weighting, and agglomerative clustering. Weighting (1.95) ranks first, followed by pruning (2.02) and clustering (2.03). Differences in rank are not significant. Summary statistics confirm the ranking with average Covering between $69-72\%$ and standard deviations of $21-25\%$. Here again, we find that the specific implementation of CP selection / ensembling accounts for only small differences in performance. We choose the best-ranking approach, CP weighting.

5.3 Comparative Evaluation

We evaluated the selected variants from three MTSS approaches and the univariate challenge baselines on the private HAS challenge data split (125 TS) to determine the best performance. Figure 7 (left) displays the mean ranks on Covering score. Distance averaging (2.74) achieves the best results, followed by CP weighting (3.22), profile averaging (3.31), PCA (3.66), univariate BinSeg (4.87), ClaSP (5.06), and FLUSS (5.15). The differences in rank for the top-3 approaches are not significant. However, all multivariate variants significantly outperform the univariate baselines. Distance averaging wins (or ties) for 53 TS, followed by profile averaging (38), CP weighting (35), PCA (33), BinSeg / ClaSP (12), and FLUSS (11). The counts do not add up to 125 due to ties. The multivariate variants only insignificantly outperform the univariate baselines for data sets with one segment (16 instances) or two segments (13). For three or more segments (96), the results align with the global ranking in Figure 7 (left).

Regarding summary statistics (Figure 7 right), distance averaging scores the highest Covering of $73 \pm 20\%$ in mean and 77% in median, with minimal differences to the 2nd and 3rd place, but a huge advance of at least 21 percentage points in mean to the challenge baselines. Similarly, distance averaging outperforms the other approaches in pairwise comparisons in at least 52% of cases.

As an example, Figure 8 illustrates the ClaSP profiles for PCA, distance averaging, and profile averaging (bottom-3) for the MTS from Figure 1. Extracted CPs are shown as red bars. Distance averaging shows peaks, captures all transitions with CPs and introduces a false positive during the train ride, possibly due to misleading magnetometer readings showing deflections throughout this

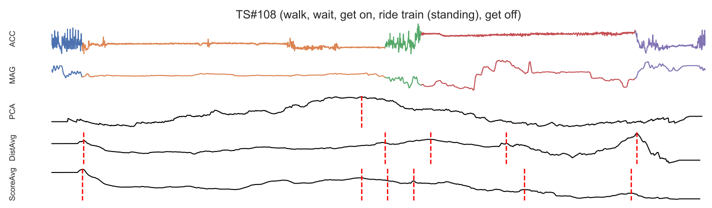


Fig. 8. Top-2: X-axis acceleration and magnetometer from Figure 1 (activities are differently coloured). Bottom-3: ClaSP profiles created using PCA, distance and profile averaging. CPs are illustrated as red bars. Distance averaging shows best results.

activity. PCA and profile averaging result in shallow profiles with missing CPs or more false positives. This demonstrates that distance averaging is interpretable for human inspection and has the best performance.

In conclusion, we find that distance averaging is the most promising approach for offline MTSS with ClaSP on the challenge data. It outperforms, yet not significantly, CP weighting and profile averaging on the private split and other model aggregation methods on the public split. It also achieves the highest average Covering score and sets a new first place on the challenge data (private split) with an F1-score of 52%, compared to CP selection with pruning at 51.5% (according to [4]). Overall, this evaluation suggests that aggregating intermediate representations of ClaSP is more effective than reducing MTS or CPs directly. However, the performance results indicate room for improvement.

6 Conclusion

In this paper, we studied and evaluated three categories to extend the ClaSP method to the multivariate setting. Dimensionality reduction and CP selection / ensembling have the advantage of being independent of the ClaSP method and can be used to extend other UTSS algorithms to the multidimensional setting. Model aggregation using distance averaging is specific to ClaSP, but leads to the best-scoring and interpretable results, setting a new state of the art on the HAS challenge data sets. Hence, we recommend using this ClaSP extension when applying the method to multivariate sensor data.

However, the performance improvement of distance averaging is small and statistically insignificant compared to CP ensembling using weighting and profile averaging. Future work should empirically investigate autoencoders for dimensionality reduction, the scalability and runtime of MTSS procedures, and see if they transfer to other domains, such as medical condition monitoring or IoT.

References

1. O. D. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE Communications Surveys & Tutorials*, vol. 15, pp. 1192–1209, 2013.

2. L. Zhou, E. Fischer, C. M. Brahms, U. Granacher, and B. Arnrich, “Duo-gait: A gait dataset for walking under dual-task and fatigue conditions with inertial measurement units,” *Scientific Data*, vol. 10, no. 1, p. 543, 2023.
3. M. A. R. Ahad, A. D. Antar, and M. Ahmed, “Iot sensor-based activity recognition - human activity recognition,” in *Intell. Syst. Ref. Libr.*, 2021.
4. A. Ermshaus, P. Schäfer, A. Bagnall, T. Guyet, G. Ifrim, V. Lemaire, U. Leser, C. Leverger, and S. Malinowski, “Human activity segmentation challenge @ ecml/pkdd’23,” in *AALTD@ECML/PKDD*, 2023.
5. A. Ermshaus, S. Singh, and U. Leser, “Time series segmentation applied to a new data set for mobile sensing of human activities,” in *EDBT/ICDT Workshops*, 2023.
6. C. Truong, L. Oudre, and N. Vayatis, “Selective review of offline change point detection methods,” *Signal Processing*, vol. 167, p. 107299, 2020.
7. S. Gharghabi, C.-C. M. Yeh, Y. Ding, W. Ding, P. R. Hibbing, S. R. LaMunion, A. Kaplan, S. E. Crouter, and E. J. Keogh, “Domain agnostic online semantic segmentation for multi-dimensional time series,” *DMKD*, vol. 33, pp. 96 – 130, 2018.
8. A. Ermshaus, P. Schäfer, and U. Leser, “Clasp: parameter-free time series segmentation,” *DMKD*, vol. 37, pp. 1262–1300, 2023.
9. H. Matsuyama, K. Hiroi, K. Kaji, T. Yonezawa, and N. Kawaguchi, “Ballroom dance step type recognition by random forest using video and wearable sensor,” in *UbiComp/ISWC*, pp. 774–780, 2019.
10. G. Harańczyk, “Change points detection in multivariate signal applied to human activity segmentation,” in *AALTD@ECML/PKDD*, 2023.
11. T.-J. Huang, Q.-L. Zhou, H.-J. Ye, and D.-C. Zhan, “Change point detection via synthetic signals,” in *AALTD@ECML/PKDD*, 2023.
12. A. Ermshaus, P. Schäfer, and U. Leser, “Raising the class of streaming time series segmentation,” *PVLDB*, vol. 17, no. 8, pp. 1953–1966, 2024.
13. Multivariate ClaSP Code, Extended Experiments and Raw Results. https://github.com/ermshaua/multivariate-clasp, 2024.
14. A. Ermshaus, P. Schäfer, and U. Leser, “Window size selection in unsupervised time series analytics: A review and benchmark,” *AALTD@ECML/PKDD*, 2022.
15. C. Wang, K. Wu, T. Zhou, and Z. Cai, “Time2state: An unsupervised framework for inferring the latent states in time series data,” *PACMMOD*, vol. 1, no. 1, pp. 1–18, 2023.
16. A. Sadri, Y. Ren, and F. D. Salim, “Information gain-based metric for recognizing transitions in human activities,” *PMC*, vol. 38, pp. 92–109, 2017.
17. S. Deldari, D. V. Smith, A. Sadri, and F. D. Salim, “Espresso: Entropy and shape aware time-series segmentation for processing heterogeneous sensor data,” *IMWUT*, vol. 4, pp. 77:1–77:24, 2020.
18. Y. Tanaka, K. Iwamoto, and K. Uehara, “Discovery of time-series motif from multi-dimensional data based on mdl principle,” *Mach. Learn.*, vol. 58, pp. 269–300, 2005.
19. I. K. Fodor, “A survey of dimension reduction techniques,” tech. rep., LLNL, Livermore, USA, 2002.
20. Y. Zhu, Z. Schall-Zimmerman, N. S. Senobari, C.-C. M. Yeh, G. J. Funning, A. A. Mueen, P. Brisk, and E. J. Keogh, “Exploiting a novel algorithm and gpus to break the ten quadrillion pairwise comparisons barrier for time series motifs and joins,” *KAIS*, vol. 54, pp. 203–236, 2017.
21. C.-C. M. Yeh, N. Kavantzas, and E. Keogh, “Matrix profile vi: Meaningful multi-dimensional motif discovery,” in *ICDM*, pp. 565–574, IEEE, 2017.
22. J. Demšar, “Statistical Comparisons of Classifiers over Multiple Data Sets,” *JMLR*, vol. 7, pp. 1–30, 2006.