A Sliding Window Filter for Time Series Streams

Gordon Lesti[1] and Stephan Spiegel[2]

[1] Technische Universität Berlin, Straße des 17. Juni 135, 10623 Berlin, Germany
gordon.lesti@campus.tu-berlin.de
[2] IBM Research Zurich, Säumerstrasse 4 , 8803 Rüschlikon, Switzerland
tep@zurich.ibm.com

Abstract. The ever increasing number of sensor-equipped devices comes along with a growing need for data analysis techniques that are able to process time series streams in an online fashion. Although many sensor-equipped devices produce never-ending data streams, most real-world applications merely require high-level information about the presence or absence of certain events that correspond to temporal patterns. Since online event detection is usually computational demanding, we propose a sliding window filter that decreases the time/space complexity and, therefore, allows edge computing on devices with only few resources. Our evaluation for online gesture recognition shows that the developed filtering approach does not only reduce the number of expensive dissimilarity comparison, but also maintains high precision.

Keywords: Internet of Things, Time Series Streams, Sliding Window Technique, Online Event Detection, Computational Complexity

1 Introduction

As time goes by things change, and those who understand change can adapt accordingly. This basic principle is also reflected in today's digital society, where sensor-equipped devices measure our environment and online algorithms process the generated data streams in quasi real-time to inform humans or cognitive systems about relevant trends and events that impact decision making.

Depending on the domain researchers either speak about events, patterns, or scenes that they aim to detect or recognize in time series, sensor, or data streams. Applications range from event detection for smart home control [17] over frequent pattern mining for engine optimization [16] to scene detection for video content [1] and gesture recognition for human-computer interaction [11].

Commonly online algorithms for data streams employ the popular sliding window technique [8], which observes the most recent sensor measurements and moves along the time axis as new measurements arrive. Usually each window is examined for a set of predefined events, which requires the comparison of the current time series segment and all preliminary learned instances of the relevant temporal patterns. In general, the pairwise dissimilarity comparisons of temporal patterns are performed by time series distance measures [18].

The time and space complexity of the sliding window technique increases with decreasing step size as well as growing window size, measurement frequency, and number of preliminary learned instances. High computational demand and memory usage is especially problematic for embedded systems with only few resources [9,19], which applies to the greatest part of sensor-equipped devices within the typical Internet of Things (IoT) scenario.

Our aim is to reduce the number of computational expensive dissimilarity comparisons that are required by the sliding window technique. To this end we propose a sliding window filter [10], which is able to decide whether the current window should be passed to a time series classifier or not. Although the filter could be considered as a binary classifier itself, it merely employs statistical measures with linear complexity and, thereby, avoids using computationally more expensive dissimilarity comparisons in many cases. Our approach to mitigate the computational complexity of event detection in data streams is different from other techniques in that we refrain from accelerating time series distance measures [13,15] or reducing dataset numerosity [20].

We demonstrate the practical use of our proposed sliding window filter for gesture recognition in continuous streams of accelleration data [10,11], where a great amount of the necessary but expensive Dynamic Time Warping (DTW) distance calculations [7] is replaced by less demanding statistical measures, such as the complexity estimate [2] or sample variance [3]. Our experimental results show that the number of DTW distance calculations can be cut in half, while still maintaining the same high gesture recognition performance.

The rest of the paper is structured as follows. Chapter 2 introduces background and notation. Chapter 3 and 4 introduce and evaluate our proposed sliding window filter. We conclude with future work in Chapter 5.

2 Background and Notation

This section gives more background on the sliding window technique [8], DTW distance measure [7], and time series normalization [4], which are fundamental building blocks of our conducted online gesture recognition study [10]. Table 1 introduces the notation that we use for formal problem description.

Symbol	Description
Q	a time series of size n with $Q = (q_1, q_2, \ldots, q_i, \ldots, q_n)$
$Q[i, j]$	a subsequence time series of Q with $Q[i, j] = (q_i, q_{i+1}, \ldots, q_j)$
t	the current time
μ	the mean of a time series Q
σ	the standard deviation of a time series Q
η, z	two different time series normalizations

Table 1. Notation used for formal problem description.

2.1 Sliding Window Technique

Given a continuous time series stream Q, the sliding window technique examines the w most recent data points and moves s steps along the time axis as new measurements arrive, where w and s are referred to as window and step size. This technique has the advantage that it does not need to store the never-ending stream of data, but it also implies that measurements can only be considered for further data analysis as long as they are located within the current window.

In most applications, each window is passed to a data processing unit, which performs some kind of time series classification, clustering, or anomaly detection. For example in online gesture recognition [10], one can employ a nearest neighbor classifier, which compares each window to a training set of preliminary learned time series instances. In case that the current window $Q[t-w, t]$ is similar to one of the known gestures, where similar means that the time series distance falls below a certain threshold, a corresponding action can be triggered. A popular distance measure for gestures [11] and other warped time series is described in the following subsection.

2.2 Dynamic Time Warping

Dynamic Time Warping (DTW) is a widely used and robust distance measure for time series, *allowing similar shapes to match even if they are out of phase in the time axis* [7]. Traditionally DTW computes a full distance matrix to find an optimal warping path, where possible nonlinear alignments between a pair of time series include matches of early time points of the first sequence with late time points of the second sequence. To prevent pathological alignments, the size of the warping window can be constraint, for instance, by the Sakoe-Chiba band [12] or the Itakura parallelogram [6]. Figure 1 illustrates the DTW distance measure using a Sakoe-Chiba band of 10%, where the percentage of the warping window refers to the length of the compared time series.

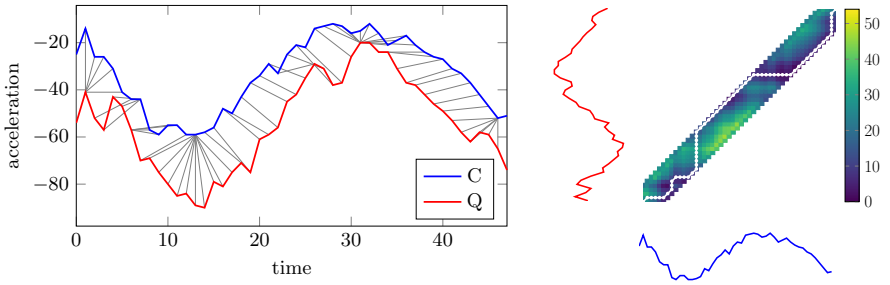

Fig. 1. Time series Q and C compared by DTW with a Sakoe-Chiba band of 10% time series length. The left plot illustrates the nonlinear alignment between the two sequences and the right plot shows the optimal warping path within the specified band.

2.3 Time Series Normalization

Literature on time series mining [5,18] suggests to normalize all (sub-)sequences before measuring their pair-wise dissimilarity by means of a distance measure. There are multiple ways to normalize time series, where two common techniques [4] are compared in this study.

Given is a time series $Q = (q_1, \ldots, q_n)$ of length n, we can compute its mean μ and standard deviation σ as followed:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} q_i \qquad\qquad \sigma = \frac{1}{n}\sum_{i=1}^{n}(q_i - \mu)^2$$

Having defined the mean μ and standard deviation σ, we can normalize each data point q_i (with $1 \leq i \leq n$) of a time series $Q = (q_1, \ldots, q_n)$ in one of the two following ways [4]:

$$\eta(q_i) = q_i - \mu \tag{1}$$

$$z(q_i) = \frac{q_i - \mu}{\sigma} \tag{2}$$

Equation 2 is commonly known as the Z-score. For the sake of simplicity we refer to η and z normalization [4] for the rest of the paper.

3 Filtering Approach

This section does not only explain the concept of our proposed filtering approach, but also describes how to integrate our filter into the well-known and widely-used sliding window technique, as shown in Figure 2.

In general, the sliding window filter considers the most recent measurements in a data stream. The considered measurements are usually passed to a classifier, which aims at categorizing the current time series subsequence. In case that the current subsequence was assigned to a known category or class, a corresponding action is triggered and the next non-overlapping window, w steps along the time axis, is examined. If the current subsequence just contains noise and no category was assigned, the next overlapping window, s steps along the arrow of time, is processed. The main limitation of this traditional sliding window technique is its computational complexity, which increases with growing window size, shrinking step size, higher sample rate, and larger training set.

For instance, given a data stream of length l=10090, a window size of w=100, and a step size of s=10, we need to classify $(l - (w - s))/s = 1000$ windows. Moreover, assuming 20 training time series, classifying 1000 windows by means of the nearest neighbor approach requires exactly $20 * 1000 = 20K$ dissimilarity comparisons. In case that we employ unconstrained DTW as time series distance measure, we need to compute $20K$ full warping matrices, each of them containing $w * w = 10K$ cells, resulting in a total amount of $200M$ distance operations.

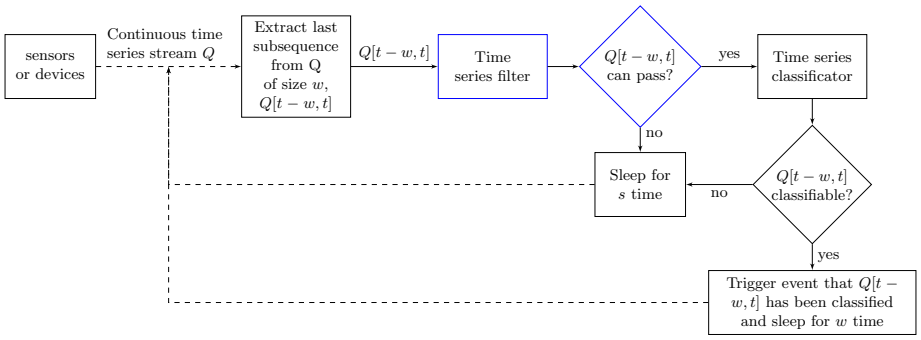

Fig. 2. Flowchart of sliding window technique with filter, highlighted in blue. The current time is denoted by t. Window and step size are denoted by w and s respectively.

In order to reduce the large number of computational expensive dissimilarity comparisons, we propose to employ a sliding window filter, which is capable of separating signal from noise, only passing promising time series subsequences to the classifier. In that sense, the proposed filter can also be considered as a binary classifier, which prunes windows that are likely to be noise and forwards subsequences that exhibit similar features as the training time series. Extracting characteristic time series features that can be used as a filter criterion should ideally exhibit linear complexity, because we aim at replacing more expensive dissimilarity comparisons. Suitable filter candidates include statistical measures, such as the sample variance [3] and length normalized complexity estimate [2], explained in more detail below.

Given is a time series $Q = (q_1, \ldots, q_n)$ with length n, we can define the sample variance (VAR) and length normalized complexity estimate ($LNCE$) as follows:

$$VAR(Q) = \frac{1}{n}\sum_{i=1}^{n}(q_i - \mu)^2$$

$$LNCE(Q) = \frac{1}{n-1}\sqrt[2]{\sum_{i=1}^{n-1}(q_i - q_{i+1})^2}$$

Having defined the above statistical measures, we are able to compute the VAR and $LNCE$ for all training time series and, subsequently, use the resulting range of statistical values to learn an appropriate filter interval. During testing, each window that exhibits a measured value within the learned interval is passed to the classifier or pruned otherwise. In order to avoid excessive pruning of relevant windows, we further more introduce a multiplication factor, which allows us to expend the interval boundaries by a certain percentage.

In general, we aim at designing a filter with high precision and recall. In our case, precision is the ratio between the number of relevant windows that were

passed to the classifier (true positives) and the number of all windows that were passed to the classifier (true positives and false positives). Consequently, recall is the ratio between the number of relevant windows that were passed to the classifier (true positives) and the number of all relevant windows (true positives and false negatives). An exhaustive evaluation of our proposed sliding window filter in dependence of all model parameters is presented in the next section.

4 Evaluation

This chapter describes the data aggregation in Section 4.1, data preparation in Section 4.2, experimental setup in Section Section 4.3, and used performance measures in Section 4.4, before presenting our results in Section 4.5.

4.1 Data Aggregation

We employed a Wii Remote^{TM} Plus controller to record different gestures for multiple users. Each user performed 8 gestures, first in a controlled environment to record clean training samples and afterwards in noisy environment to record a test time series stream, which includes all predetermined gestures as well as acceleration data that corresponds to other physical activities. All records are available for download on our project website [10]. A sample record containing both training and test gestures is illustrated in Figure 3.

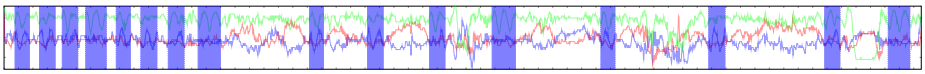

Fig. 3. An sample record of 98 seconds length, containing 8 training gestures at the very beginning, directly followed by a test time series stream that comprises the same 8 gestures mixed with acceleration data of various physical activities. Gestures are highlighted by blue rectangles and are also marked in the recorded data. Note that the length of gestures can vary for training and test set as well as for different records.

4.2 Data Preparation

All of our data records are resampled and quantized before further analysis. In general, dimensionality and cardinality reduction of time series is performed to ease and accelerate data processing by providing a more compact representation of equidistant measurements [11].

Resampling: The recorded acceleration data was resampled by means of the moving average technique, using a window size of 50 ms and step size of 30 ms.

Acceleration data (a) in $\frac{dm}{s^2}$	**Converted value**
$a > 200$	16
$100 < a < 200$	11 to 15 (five levels linearly)
$0 < a < 100$	1 to 10 (ten levels linearly)
$a = 0$	0
$-100 < a < 0$	-1 to - 10 (ten levels linearly)
$-200 < a < -100$	-11 to - 15 (five levels linearly)
$a < -200$	-16

Table 2. Conversion of recorded acceleration data from $\frac{dm}{s^2}$ scale to integer values.

Quantization: The resampled records were then converted into time series with integer values between -16 and 16, such as suggested in related work [11] and summarized in table 2.

4.3 Experiment

The proposed sliding window filter has several model parameters that need to be carefully tuned in order to achieve optimal performance. Depending on the application domain we need to select an appropriate window and step size, time series normalization, dissimilarity threshold, and filter criterion. In the following we describe all parameter settings that were assessed in our empirical study:

- The **window size** determines the number of most recent measurements contained in the examined time series subsequences. We tested four different sizes that were learned from the training gesture, including **min**, **max**, and **avg** length as well as the **mid**-point of the range.

- The **step size** defines the gap between consecutive time series windows. As default setting we use one tenth of the window size.

- For online gesture recognition we employ the nearest neighbor classifier in combination with the DTW distance, where we evaluate 34 different Sakoe-Chiba **band** sizes, ranging from 1 % to 100 %. Prior to pair-wise comparing sliding windows and training gestures, the corresponding time series should be normalized. We evaluate η, z, and no **normalization**.

- The dissimilarity **threshold** defines the time series distance at which a sliding window and a training gesture are considered to belong to the same class. We determine the threshold for an individual class by measuring the distances between all samples of that particular class and all instances of other classes. In our empirical study we evaluate the threshold influence for: (i) one half of the minimum distance - **HMinD**, (ii) one half of the average distance - **HAvgD**, and (iii) one half of the midpoint distance - **HMidD**.

– The **filter criterion** is an essential part of our proposed approach. In our empirical study we evaluate the performance of the two filter criteria, namely the sample variance **VAR** and the length normalized complexity estimate **LNCE** of a time series. Both filters are tested with different factors that increase the size of the filter interval from 100 % to 300 %.

Figure 4 visualizes the online gesture recognition results for a sample time series stream processed by our proposed sliding window filter, after selecting the above described model parameters with help of the recorded training gestures.

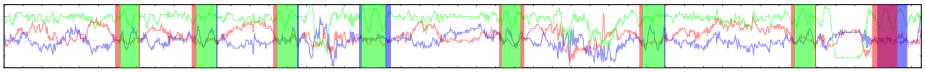

Fig. 4. Visualized results of online gesture recognition for a sample time series stream. We highlight true positives in green, false positives in red, false negatives in blue, and true negatives in transparent. Although we see short false detection intervals before or after true positives, seven out of eight gestures were assigned to the correct class label.

4.4 Performance Measures

Since our proposed sliding window filter is tested on time series streams that contain various different gestures, we need to treat the described online gesture recognition challenge as multi-class problem. Common performance measures for multi-class problems are $Precision_\mu$, $Recall_\mu$ and $F_\beta score_\mu$ [14]:

$$Precision_\mu = \sum_{i=1}^{l} tp_i \Big/ \sum_{i=1}^{l} (tp_i + fp_i)$$

$$Recall_\mu = \sum_{i=1}^{l} tp_i \Big/ \sum_{i=1}^{l} (tp_i + fn_i)$$

$$F_\beta score_\mu = (\beta^2 + 1) Precision_\mu Recall_\mu \Big/ \beta^2 Precision_\mu + Recall_\mu$$

where β is usually set to one and l denotes the number of classes that require separate computation of true positives (tp), false positives (fp), and false negatives (fn). These multi-class performance measures allow us to compare and rank the results for different parameter settings. For our evaluation we employ the $F_1 score_\mu$, which weights $Precision_\mu$ and $Recall_\mu$ equally.

4.5 Results

In order to evaluate the influence of all model parameters that were described in Section 4.3, we performed a total number of 28152 experiments. Figure 5(a)

illustrates the $Precision_\mu$ and $Recall_\mu$ values for all test runs. A top performance of around 0.7384 F_1score_μ was achieved by parameter configurations that used η time series normalization, DTW with a Sakoe-Chiba band of about 18 % time series length, *mid* window size, and *HAvgD* for threshold determination.

Given the best parameter configuration, we investigated the influence of the individual parameters by changing only one at a time and fixing the others, see Figure 5(b,c,d). As shown in Figure 5(e), we also evaluated the performance with VAR, $LNCE$, and no filter. The best results for each individual gesture is shown in Figure Figure 5(f). Further tests on the applicability of the sliding window filter as well as our interpretation of the results are presented below.

Normalization: The influence of the time series normalization is illustrated in Figure 5(b). We compare η, z, and no normalization, with *mid* window size and *HAvgD* dissimilarity threshold. The best F_1score_μ was achieved by means of the η normalization, which corresponds to the data points shown in the magnifying glass. The point cloud in the lower left corner of plot 5(b) are parameter settings with rather small warping band.

Warping Band: The influence of the Sakoe-Chiba band in combination with the DTW distance is shown below in Figure 6. For this experiment we selected only the dominating parameter settings, with η normalization, *mid* window size, and and *HAvgD* dissimilarity threshold. The best F_1score_μ was achieved with a band with of 18 % time series length.

Dissimilarity Threshold: We evaluate three different ways of determining a dissimilarity threshold, namely *HMinD*, *HAvgD*, and *HMidD*. For our comparison in Figure 5(c), we used η normalization, *mid* window size, and a warping band of 18 % time series length. The best F_1score_μ was achieved by means of *HAvgD*, shortly followed by the *HMidD* approach. Comparatively high $Precision_\mu$ values were given by *HMinD* threshold.

Window Size: The influence of the window size determination approach is shown in Figure 5(d). We compare *min*, *max*, *avg*, and *mid* window size, with η normalization, *HAvgD* dissimilarity threshold, and a warping and of 18 % time series length. The highest $Precision_\mu$, $Recall_\mu$ and F_1score_μ was achieved by the *mid* window size, shortly followed by the *avg* window size. Figure 5(d) furthermore suggests to refrain from using *max* and *min* window size determination.

Filtering Approach: Given the optimal parameter setting that was determined in the previous experiments, we are now in the position to assess the influence of the filtering approach. Figure 5(e) shows the performance with VAR, $LNCE$, and no filter. Twenty simulations are reaching a F_1score_μ value greater or equal to 0.7. Interestingly, top performance was achieved with and without filter. This lead is to the question of computational complexity, which is answered in following paragraph.

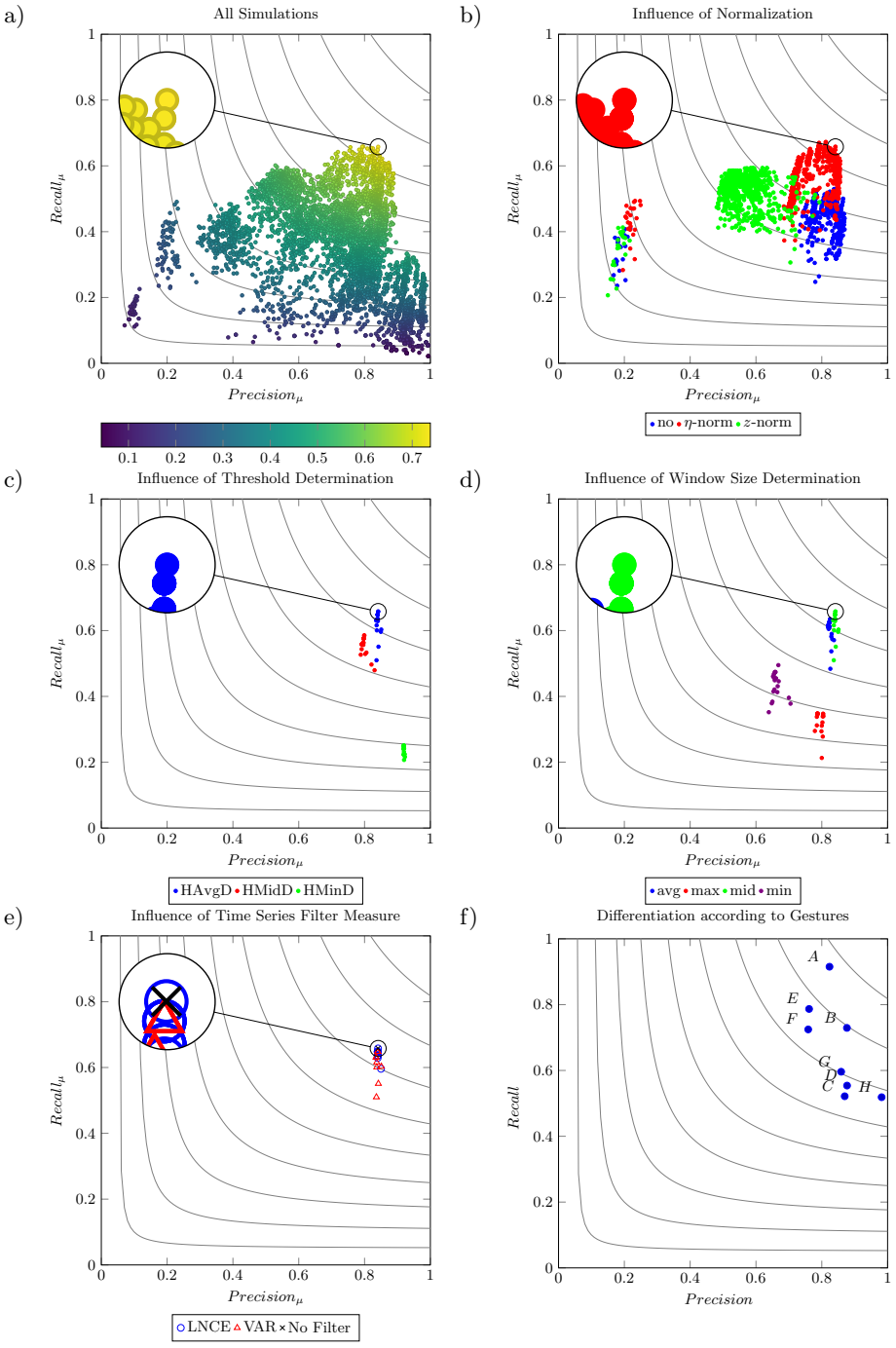

Fig. 5. $Precision_\mu$ and $Recall_\mu$ plots illustrating the performance influence of the individual model parameters. The magnifying glass is focusing on the results with the highest F_1score_μ. Gray lines indicate the F_1score_μ distribution in $\frac{1}{10}$ steps.

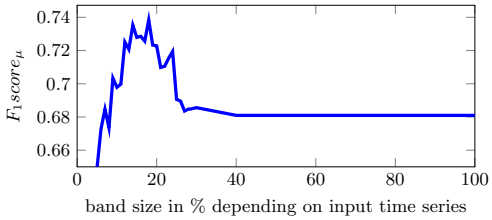

Fig. 6. Performance with varying Sakoe-Chiba band width.

Filter Interval: The filter interval does not only influence the resulting F_1score_μ, but also the amount of time series dissimilarity comparisons. Figure 7 illustrates the influence of the interval size in respect to performance and computational demand. With an appropriate filter interval of about 200 % we are able to reduce the number of dissimilarity comparisons by one half, while still achieving relatively high performance values.

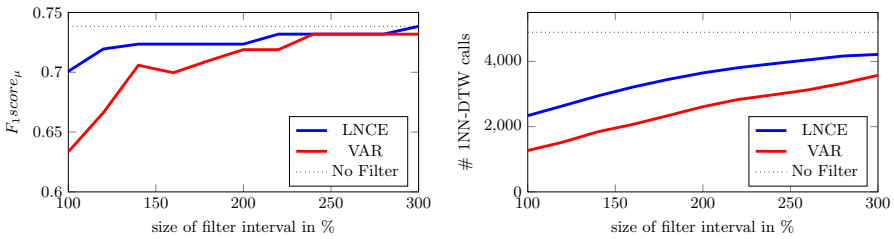

Fig. 7. Influence of filter interval on the F_1score_μ (left) and the amount of 1NN-DTW dissimilarity comparisons (right). An optimal tradeoff between performance and computational demand is achieved at approximately 150 % to 200 % filter interval.

Individual Gestures Finally, we have tested the performance for each of the examined gestures separately. Figure 5(e) shows the best *Precision* and *Recall* values that were achieved for each gesture. The results demonstrate that some gestures are easier to recognize than others.

5 Conclusion and Future Work

In this work we have proposed a novel sliding window filter for more efficient event detection in time series streams, which replaces computational expensive dissimilarity comparisons by less demanding statistical measures. Furthermore, we have demonstrated that the developed filter is able to recognize gestures in continuous streams of acceleration data with high accuracy, while cutting the number of distance calculations in half.

Possible applications do not only include event detection on mobile device with few hardware resources, but also distributed sensor networks with limited bandwidth that communicate high level information instead of transferring raw data. In future work we plan to investigate a larger variety of statistical measures that exhibit favorable filtering properties for online gesture recognition as well as data streams found in other domains.

References

1. E. Acar, S. Spiegel, and S. Albayrak. MediaEval 2011 Affect Task: Violent Scene Detection combining audio and visual Features with SVM. CEUR Workshop, 2011.
2. G. Batista, X. Wang, and E. J. Keogh. A complexity-invariant distance measure for time series. ICDM, 2011.
3. T. F. Chan, G. H. Golub, and R. J. LeVeque. Algorithms for computing the sample variance: Analysis and recommendations. The American Statistician,1983.
4. G. Das, K.-I. Lin, H. Mannila, G. Renganathan, and P. Smyth. Rule Discovery from Time Series. KDD, 1998.
5. H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. VLDB Endowment, 2008.
6. F. Itakura. Minimum prediction residual principle applied to speech recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing, 1975.
7. E. Keogh. Exact indexing of dynamic time warping. VLDB Endowment, 2002.
8. E. Keogh, S. Chu, D. Hart, and M. Pazzani. Segmenting time series: A survey and novel approach. Data mining in Time Series Databases, 2004.
9. S. Kratz, M. Rohs, K. Wolf, J. Mueller, M. Wilhelm, C. Johansson, J. Tholander, J. Laaksolahti. Body, movement, gesture and tactility in interaction with mobile devices. MobileHCI, 2011.
10. G. Lesti and S. Spiegel. The sliding window filter for time series streams website. https://gordonlesti.com/a-sliding-window-filter-for-time-series-streams/
11. J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan. uWave: Accelerometer-based personalized gesture recognition and its applications. PerCom, 2009.
12. H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. Transactions on acoustics, speech, and signal processing, 1978.
13. D. Sart, A. Mueen, W. Najjar, E. Keogh, and V. Niennattrakul. Accelerating dynamic time warping subsequence search with GPUs and FPGAs. ICDM, 2010.
14. M. Sokolova and G. Lapalme. A systematic analysis of performance measures for classification tasks. Information Processing and Management, 2009.
15. S. Spiegel, B.-J. Jain, and S. Albayrak. Fast time series classification under lucky time warping distance. SAC, 2014.
16. S. Spiegel. Discovery of driving behavior patterns. Smart Information Systems - Advances in Computer Vision and Pattern Recognition, 2015.
17. S. Spiegel. Optimization of in-house energy demand. Smart Information Systems - Advances in Computer Vision and Pattern Recognition, 2015.
18. S. Spiegel. Time series distance measures: segmentation, classification and clustering of temporal data. TU Berlin, 2015.
19. M. Wilhelm , D. Krakowczyk, F. Trollmann, S. Albayrak. eRing: Multiple Finger Gesture Recognition with one Ring Using an Electric Field. iWOAR, 2015.
20. X. Xi, E. Keogh, C. Shelton, L. Wei, C. A. Ratanamahatana. Fast time series classification using numerosity reduction. ICML, 2006.