IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 26, 2021, accepted March 9, 2021, date of publication March 17, 2021, date of current version March 25, 2021.

*Digital Object Identifier 10.1109/ACCESS.2021.3066686*

# Towards Smart Data Selection From Time Series Using Statistical Methods

**AMAIA GIL[1,2], MARCO QUARTULLI[1], IGOR G. OLAIZOLA[1], AND BASILIO SIERRA[2]**

[1]Vicomtech Foundation, Basque Research and Technology Alliance (BRTA), 20009 Donostia-San Sebastián, Spain
[2]Department of Computer Sciences and Artificial Intelligence, University of the Basque Country (UPV/EHU), 20018 Donostia-San Sebastián, Spain

Corresponding author: Amaia Gil (agil@vicomtech.org)

This work was supported in part by the 3KIA Project through ELKARTEK, Basque Government.

**ABSTRACT** Transmitting and storing large volumes of dynamic / time series data collected by modern sensors can represent a significant technological challenge. A possibility to mitigate this challenge is to effectively select a subset of significant data points in order to reduce data volumes without sacrificing the quality of the results of the subsequent analysis. This paper proposes a method for adaptively identifying optimal data point selection algorithms for sensor time series on a window-by-window basis. Thus, this contribution focuses on quantifying the effect of the application of data selection algorithms to time series windows. The proposed approach is first used on multiple synthetically generated time series obtained by concatenating multiple sources one after the other, and then validated in the entire UCR time series public data archive.

**INDEX TERMS** Data selection, machine learning, optimization, time series.

## I. INTRODUCTION

Fine grained, high temporal resolution sensor dynamic data is often useful for short-term forecasting and visualization [1]. However, communication latency, bandwidth constraints, high energy consumption and storage requirements for such data can be problematic [2]. Reducing the amount of data to be transmitted can help control latency time and save in energy consumption and storage [3].

A key challenge in the setup of point selection methodologies is reducing the size of the transmitted data without sacrificing its quality. A natural solution is to compress the data at the sensing devices, monitoring in real time the error introduced by this process. When adaptive point selection strategies are used [4], the objective is to select a subset of data points with a well-defined number of items to be transmitted periodically. Then, the effect of this compression methodology on subsequent data analysis and exploitation processes can be studied, for instance considering the difference between the recovered and the compressed versions of the data for a given original time series.

Blalock *et al.* [5] describe desirable properties of the compression algorithms:

1) Minimal buffering: on devices with small memory capacities, only small time windows can be used before data is compressed. Furthermore, large buffering can add unacceptable latency.
2) High decompression speed: decompression of data in order to recover the time series for other parts of the service such as visualization and machine learning applications needs to be quick.
3) Losslessness: noise and oversampling of data vary with time and depend on application. The compression is seen as a preprocessing step that is application specific. Using lossless compression algorithms ensure that the data could not depend on previously defined preprocessing strategies.

On the one hand, most work on compressing time series has focused on lossy techniques. Classical approaches for data compression include Fourier transforms [6], wavelets transforms [7], symbolic representation [8] and piecewise regression [1], [9].

The Fourier transform is a tool widely used for spectral analysis, signal filtering and compression. This transformation is adequate for analyzing the components of a stationary signal, as the sinusoidal components are propagated in all the time domain. For non-stationary signals the Fourier transform analysis is not appropriate because it is not able to

The associate editor coordinating the review of this manuscript and approving it for publication was Gianluigi Ciocca.

maintain any localized information of a signal. Wavelets are oscillations that decay quickly allowing an adequate analysis of non-stationary signals [10]. A common application of these transformations is signal compression. A threshold is defined and components with smaller value than the threshold defined are removed. Thus, after reconstruction by inversion formulas the signal maintains its original shape [11].

Symbolic representation approaches are designed to preserve enough information about the time series to support indexing or specific data mining algorithms, rather than to compress the time series per se. In order to change to symbolic compression, dimensionality reduction is usually applied by a window aggregation function such as piecewise aggregate approximation. Later, the variable values are normalized. Defining the aggregate functions implies not being able to reconstruct the signal easily, which amounts to losing the original measured data points. Finally, a symbol is selected depending on the range of values that it maps to.

Piecewise regression techniques divide a time series into fixed-length or variable length intervals and describe them using regression functions. As for regression functions, all types of functions can be used in principle. However, low-order polynomial functions, particularly constant and linear functions, can be estimated efficiently and are used frequently [12].

Yang *et al.* [13] propose using clustering techniques to group time series by similarity. A workload distribution strategy can be taken using this cluster division saving time in processing the compression. Then, each of the time series is compressed using autoregressive models.

Classical compression techniques reduce the volume of data by using transformations, regression models or aggregations functions. The result of the transformations, the parameters of the regression models or the symbolic representation of the aggregated values are stored to represent the signal. None of the data points measured are transmitted and the signal representation is dependent on the efficacy and adequateness of the compression methodology used.

On the other hand, lossless compression techniques use binary encoding for the representation. Pelkonen *et al.* [14] propose a compression algorithm that maintains the full representation of time series. It compresses separately the timestamp and the value of the data point. The timestamp part employs an efficient delta-of-delta encoding, while the measurement part uses a XOR'd floating point approach. The strategy of Blalock *et al.* [5] employs the predictability of a data point to obtain an effective encoding of the difference between the predicted values and the original one. This is done in order to take advantage of the correlation between continuous data samples.

Vestergaard *et al.* [15] propose a two step compression technique that allows advantages in terms of random access. First, in the preprocessing step the system determines the adequate values of the input parameters of the compression technique, such as number of samples in a chunk, using part of the data for the training. Then, the time series is divided in chunks and compressed separately, implying no need to decompress complete the time series for a random access.

Lossless compresssion techniques help reducing the volume and storage of data to be transmitted and satisfy all the properties above mentioned. However, the compressed version of the signal cannot be used for visualization, control or analytic applications directly, as all the data points are saved with the same quality, only the encoding time series data has been optimized to save storage.

An alternative to compressing techniques is using point selection algorithms to reduce the data volume. These techniques aim to select the most significant or representative points. One option apart from selecting points from a regular sampled signal would be using adaptive sampling methods [16]. These approaches study the level of variance between the collected data over a certain time frame and dynamically adjust the sampling frequency of the device. Adaptive sampling approaches work well in applications where the collected time series are stationary. In the case of quickly varying data, these approaches perform poorly.

It is desirable to be able to compress time series from stochastic processes into streams with constant or limited length in order to meet memory capacity limitations. To the best of our knowledge, there has been no reported work on time series compression with rate adaptability and the ability to flexibly preserve different characteristics of interest of a given time series. In this sense, the contributions put forward by the present paper are:

1) The idea of combining different data points selection methodologies using their potential in the current signal window.
2) A definition of errors for the determination of the optimal data points selection methodology in each moment and for different characteristics of interest relative depending on the envisaged application
3) An algorithm that implements the above methodology.

The proposed approach searches, inside a defined set of point selection algorithms, the optimal solution for the actual time frame of the time series. The adequateness of the selected algorithm can be evaluated and monitored in time to guarantee the quality of the compression technique.

Even if this compression strategy is lossy, monitoring the compression allows controlling window sizes and deciding when to retrain the point selection model in order to adjust it to the current characteristics of the signal. With this approach, the above mentioned desired properties of compression systems can be controlled by the user. This process is shown in Figure 1.

This methodology has been validated with several synthetically generated time series and with all the datasets available in the UCR Time Series Classification Archive [17]. The proposed approach is capable to adapt to the dynamics of the time series effectively using different error functions.

The rest of the paper is structured as follows. Section II introduces the available point selection methods. Then,

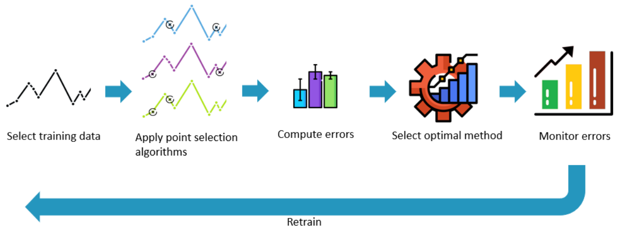


**FIGURE 1. Adaptive optimal point selection method process. First, the training data is selected and multiple point selection methods are applied. Then, the optimal algorithm is selected and introduced in the system as subsampling method. Finally, the errors are monitored in the process and if a model drift is detected, the point selection model is retrained.**

the methodology and the proposed approach are explained in section III-A. Next, an example is shown in order to demonstrate its usefulness in section III-B. Section IV provides a description of the experiments setup, whereas section V shows the results of those experiments. Finally, conclusions and future work are presented in section VI.

## II. THEORETICAL BACKGROUND

Consider a time series signal which is sampled with a constant frequency. Due to system limitations, for example with respect to memory, not all captured points can be stored, and therefore a data point selection methodology needs to be applied.

One possible classification of the data points selection algorithms is to consider the way in which those are applied [18]:

- Algorithms that work in batch mode: the data is processed in group or batches. The algorithm is used only when the batch or group is complete. This can require fewer network resources than online systems.
- Algorithms that work in online / streaming mode: when a new point arrives to the system the data points selection methodology is applied directly. A previously saved snapshot representing the (e.g. statistical) properties of previous points and the most recently received points need to be available in order to decide if the actual received point is saved.

Other possibility was proposed by Keogh *et al.* [4], a classification for data point selection algorithms based on the point selection strategy they adopt:

- Selection of the best representation of the time series with a maximum error at each point (local error) less than a certain value (*max_error*).
- Selection of the best representation of the time series with a maximum combined error by all the segments (global error) less than a given value (*max_total_error*).
- Selection of the best representation of the time series using $k - 1$ segments (or equivalently $k$ points).

In sections II-A, II-B and II-C different point selection algorithms found in the state of the art are explained using the above mentioned classification system.

### A. DATA POINTS SELECTION USING A MAXIMUM VALUE FOR LOCAL ERRORS

A review of classical data point selection methodologies based on a maximal error value in a point of the time series is described in Watson *et al.* [19]. All the strategies work in online mode and depend on a maximum error threshold (*max_error*) that should be indicated by the user. This input value is defined using background knowledge such as sensor limitations or a variable noise scale. The appropriate selection of a threshold value guarantees that no important information is lost in the data point selection procedure.

The boxcar algorithm makes a selection of a point when the current value differs from the last saved value by an amount greater than or equal to the determined maximum error threshold bound for that variable. The last processed value before the one which exceeds the limit should be saved.

The backward slope methodology projects the error bound into the future on the basis of the line formed by the previous two data points. The first data point is selected if the second value lies outside the error bound. Once a new data point is recorded, the new line and the error bound are projected into the future, repeating the same strategy.

The swinging door strategy is similar to the one considered by the backward slope algorithm, except that the error bound is based on the slope of the line between the first and the current data point. When the current data point has exceeded the error bound defined, the data point at the previous time index is selected. Then the error bound and line are recalculated and the algorithm is repeated.

Keogh *et al.* [4] propose two point selection strategies that work in an iterative mode. Therefore, a fixed buffer or window size is needed in these cases, i.e, these strategies work in batch mode.

The top-down strategy starts considering all the segments between adjacent points. Then, the algorithm continuous merging contiguous segments by removing the intermediate point, i.e., the common extreme of the contiguous segments, that adds a minimal error value from all the possibilities. This is done in a iterative way and until the *max_error* value is not exceeded.

The bottom-up strategy starts instead with a unique segment defined by the two extreme points of the time series (first and last in time index). The algorithm adds points to the selected set in an iterative. Each time the point that have the greatest error in the actual representation of segments is added in the set and the segments are recalculated. This is done until the error committed is less than the *max_error* value.

These two strategies are adapted to the specific cases detailed in sections II-B and II-C by specifying different stopping criteria.

### B. DATA POINTS SELECTION USING A MAXIMUM TOTAL ERROR VALUE

In the case of data points selection using a maximum total error value, a total error is calculated each time using an aggregation function from total errors. That value is used and points are added to the selected set while the total error value is higher that the defined limit *max_total_error*.

### C. DATA POINTS SELECTION USING MAXIMUM NUMBER OF SEGMENTS

The algorithms detailed in this section work in batch mode. In the case of data points selection using maximum number of segments, a fixed window in time series data is used as a batch and from there a maximum number of points $k$ is selected to be part of the compressed signal. The value $k$ should be defined by the user taking into account the limitations of the system, such as memory limits. The following paragraphs detail different algorithms with this objective.

The different versions of the largest triangle algorithm [20] are based on the use of the effective area of the data points: the significance of a point is indicated by the area of the triangle formed with its two adjacent points. Depending on how the adjacent points are selected or how the buckets are constructed, three different algorithms are generated.

- Largest-triangle-one-bucket (*ltob*): first, the effective areas for each point is calculated using prior and posterior data points in the time series. Then, $k$ buckets are generated splitting the time series with approximately equal number of points in each of them. From each bucket the data point with the largest effective area is selected. In order to guarantee that the first and last point of the time series are selected, extreme buckets only contain those points.
- Largest-triangle-three-buckets (*lttb*): in this case, the effective area of a point does not depend on the position of its two adjacent points as in the previous case, it takes into account all data points from previous and posterior buckets. For that, first buckets are generated in the same way as the previous version of the algorithm (each bucket with nearly equal quantity of data points, except for extreme buckets that only contain the first an the last data points). Then, the effective area of each point is calculated using the mean value of data points from the posterior bucket and the data point selected from the previous bucket. Finally, the point with largest effective area is selected in each bucket.
- Largest-triangle-dynamic: this version of the algorithm does not rely on equal size buckets, but the buckets are generated in a iterative mode, starting from default buckets (equally sized). In order to determine which bucket needs to be larger or smaller, a linear regression model is fitted with the data points in each the bucket, the last data point of the previous bucket and the first point of the posterior bucket. Then, the fitted linear model validity is measured by the sum of squared error (SSE). Later, the bucket with the maximum SSE is divided in two and the bucket containing the minimum error is merged with one of its adjacent buckets (the one with minimum error option), guaranteeing that the number of buckets continuous to be equal to the limitation of the maximum number of selected points $k$. After each iteration, as new buckets are generated, linear regression models need to be recalculated. After a certain number of iterations, when the buckets sizes have become stable (or with similar SSE values), largest-triangle-three-buckets algorithm is used to calculate the effective area of each data point, finally selecting the most meaningful data point from each bucket.

The mode-median-bucket (*mmb*) [20] algorithm uses the mode and the median values of data points in each bucket in order to select a point from it. The data points are split into buckets that contain approximately the same number of data points. Then, each bucket is studied separately. If there is a unique mode in the bucket, the leftmost corresponding data point is selected. Otherwise, the data point equal to the median value from the bucket is selected. An exception happens with the minimum and maximum values of the time series, these peak points are selected directly from the buckets that contain them, in order to guarantee the preservation of extreme data points.

The M4 (*m4a*) strategy was defined by Jugel *et al.* [21]. First, $n$ buckets are generated containing approximately equal number of points. In this case, as 4 points could be selected from each bucket, the number of buckets is equal to $n = truncate(k/4)$. Then, from each bucket the minimum and maximum values from both axis (time index and data values) are selected (hence the name M4). In some cases, the minimum or/and maximum point(s) in both axes can be represented by a unique data point, i.e., when the maximum or minimum data values occur in the minimum or maximum time index of the bucket, one point could be selected by two rules, selecting finally less quantity of points than expected initially. Bae *et al.* [22] expand the *m4a* point selection strategy for visualization services also using gradient values between adjacent columns of pixels to reduce more points.

Major extrema extraction technique proposed by Fink and Gandhi [23] consists on ranking the extrema values of the time series and selecting the most meaningful ones. These extremes would be the finally selected points for the compressed version of the time series. They considered four types of extrema: strict, left, right and flat. By strict they refer to local minimum and maximum points of the time series. Left and right are the extremes in time of a flat chunk and flat is a inner point of the flat chunk. Then, the importance of each type of extrema is calculated by the use of a distance and a positive parameter that determines the compression rate.

The simplest way to select data points from a time series with a constant sampling frequency is to pick them using a lower frequency value than the original one, i.e, selecting a point each n points (*oen*). This algorithm takes into account the maximum number of selected points ($k$) in order to select the new frequency $w$ ($w = truncate\left(\frac{length(y_o)}{k}\right)$).

Top-down and bottom-up strategies can be modified using this time the length of the selected points set equal to the number of desired selected points $k$ as stopping rule.

## III. PROPOSED APPROACH: SMART COMPRESS

In this paper, a smart data selection method based on a optimization process is proposed. The aim of this optimization problem is to select the method that minimizes the errors of the point selection process for each feature.

First, a detailed explanation of the methodology is presented in section III-A and an example of application is shown afterwards in section III-B.

### A. METHODOLOGY

The general concept of the proposed Smart Compress methodology can be described as follows:

1) First, the data point selection model is fitted using training data; in other words, the fit method selects the optimal algorithm that suits best the time series provided in the training.
2) Then, a compressed version of the signal is obtained by applying the fitted data selection method to the test data.
3) Finally, the adequateness of the data selection model is validated using the error between the original and the recovered signal from decoding the compressed signal.

Suppose there is a time window of the selected time series $y_o$ that needs to be compressed to be transmitted by a limited channel. First of all, the fitting method is used to identify the optimal data points selection method for compression. The inputs needed for the fitting method are the data points in the selected time window $y_o$ and a threshold indicating the maximum number of points that a compressed version of the signal could have ($k$). Then, another window of the same time series ($z_o$) can be used for testing the adequateness of the data points selection algorithm by the use of the method score. Finally, the optimal data points selection method is used for compressing other windows of the same time series with the predict method. This process is depicted in Figure 2.

In order to be able to compare different data point selection methodologies, the compressed version of the time series ($y_c$) should have a similar quantity of selected points after the compression strategy is applied to the original data ($y_o$). For this reason, the methods considered in the smart selection algorithm are the ones described in section II-C.

The algorithm selected as the baseline is *oen* as it is the simplest strategy that can be applied and the quality of the signal can be random in some cases. Furthermore, top-down, bottom-up, major extrema extraction and largest-triangle-dynamic algorithms are not considered in the experiments as the methods become very slow depending on the length of points in the window / buffer considered and the number of points to be selected.

The fitting process to find an optimal data points selection model could be mathematically represented as follows:

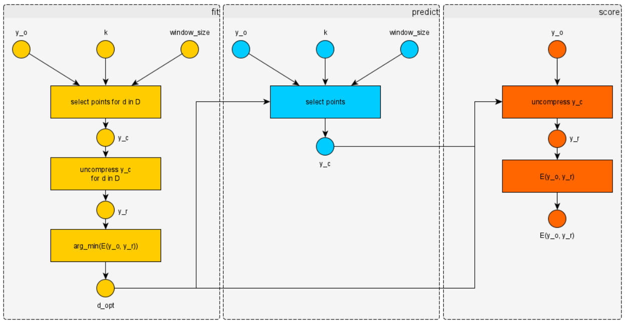


**FIGURE 2. High-level view of the Smart Compress concept. The three available methods (fit, predict and score) are shown to indicate outputs and inputs in each case. First, a training time series is used, together with the parameter *k*, to identify an optimal points selection method (yellow part of the diagram). Once the method is identified, this optimal point selection is used by the score method to validate the result (shown in orange) and by the predict method to obtain the compressed version of a time series (shown in blue).**

Suppose there is a window of the time series $y_o(t_o)$ where $t_o = [t_{o1}, t_{o2}, \ldots, t_{om}]$ and being $m$ the number of points in the selected window. Let $d$ be a data points selection method from the available methods set $D = \{ltob, lttb, m4a, mmb, oen\}$. Then, the compressed version of the time series, $y_c(t_c)$, is defined by the selected data points from $y_o$ corresponding to time indexes $t_c = [t_{c1}, t_{c2}, \ldots, t_{cn}]$. The value $n \leq k$ where $k$ is the maximum allowed quantity of $y_c(t_c)$.

From $y_c(t_c)$ the removed data points values are recovered by the use of linear interpolation method between available points of $y_c(t_c)$. The notation used to refer to the reconstructed version of the time series is $y_r$ and it is defined for time index values that where contained in the original time series signal $t_o$.

Finally, an optimization problem is defined to select the most adequate method for the signal. This optimization is represented by:

$$\underset{d \in D}{\arg\min}\, E(y_o, y_r) \tag{1}$$

The detail of the fit method just explained is described graphically in Figure 3.

Depending on the purpose of the application, the most interesting properties of the signal could be totally different. The error functions can be defined in order to maintain these properties of the signal. Different signal characteristics are listed next for three different purposes:

- In visualization applications, properties such shape of the signal, visual outliers, linear trend of data and number of peaks are important to maintain. The general visual distortion generated from the compression can be measured by absolute sum of changes, mean absolute change, mean change, mean second derivative central and complexity-invariant distance.
- In control applications, the appearance of new events (peaks), change in signal tendency or frequency are

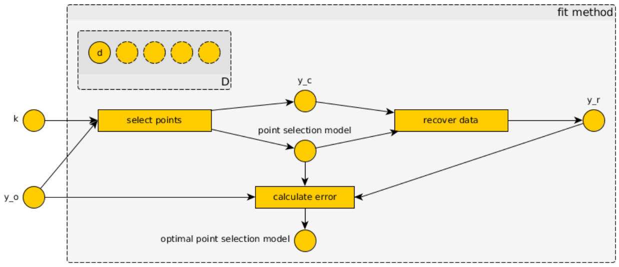


**FIGURE 3. Fit methodology detail. The $k$ parameter and the time series used for the training ($y_o$) are the inputs of the fit method. All the points selection methods available in $D$ are considered separately. From each method used, a compressed version $y_c$ and a recovered version $y_r$ of the time series are obtained. This last time series $y_r$ is compared with $y_o$, and the quality of the compression algorithm is measured by the error function. Finally, the identified optimal algorithm is saved as an inner object of the Smart Compress system for its later use by the score and predict methods.**

important. In statistical control process applications, values ratio beyond r times standard deviation, longest strike above/below mean and count elements above / below certain value need to be considered.

- For analytical proposes, outliers and statistical properties such kurtosis, maximum, mean, median, minimum, quantiles, skewness, standard deviation and variation coefficient are essential.

Four different error functions have been used in the experiments. These error functions are selected in order to measure the distortion generated due to the use of the point selection algorithm. These error functions are detailed next:

- Percentage RMS difference [24]:

$$PRD = \left(\frac{\sum(y_o(i) - y_r(i))^2}{\sum(y_o(i))^2}\right)^{1/2} \tag{2}$$

- Normalized root mean square deviation [25]:

$$NRMSD = \frac{\left(\frac{\sum(y_o(i)-y_r(i))^2}{length(y_o)}\right)^{1/2}}{max(y_o) - min(y_o)} \tag{3}$$

- Mean absolute error [26]:

$$MeAE = \frac{mean(abs(y_o(i), y_r(i))))}{max(y_o) - min(y_r)} \tag{4}$$

- Maximum absolute error [27]:

$$MaAE = \frac{max(abs(y_o(i), y_r(i))))}{max(y_o) - min(y_r)} \tag{5}$$

### B. PRELIMINARY APPLICATION EXAMPLE

The considered datasets are the ones available at the UCR Time Series Classification Archive [17]. The Archive contains 128 classification time series datasets of different types including sensor data, simulated data, motion data from several devices and health data such as electrocardiograph (ECG), electrooculography (EOG) and hemodynamic data. Depending on the dataset, either all the time series contained in it have same length, or the length varies between different time series.

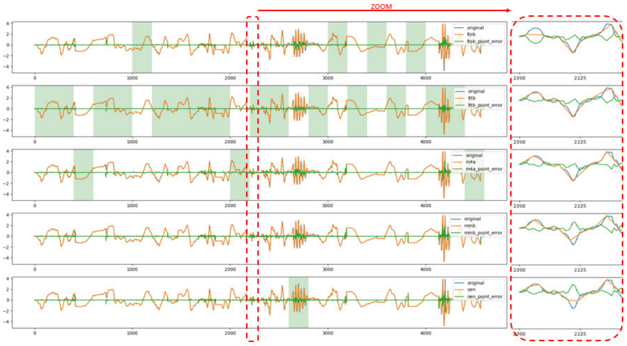


**FIGURE 4. $y_o$ (in blue), $y_r$ (in orange), $y_r - y_o$ (in green) time series from a synthetically generated time series. The optimal method selected by *MaAE* in each window is highlighted. At right, a detailed view of a representative time segment is added.**

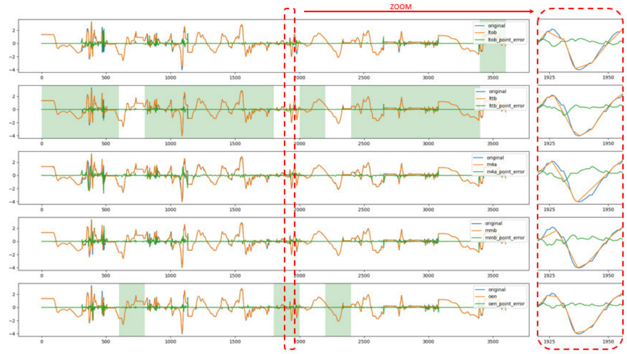


**FIGURE 5. $y_o$ (in blue), $y_r$ (in orange), $y_r - y_o$ (in green) time series from a synthetically generated time series. The optimal method selected by *MeAE* in each window is highlighted. At right, a detailed view of a representative time segment is added.**

Several synthetic time series are generated combining time series from two different datasets (AllGestureWiimoteX and UWaveGestureLibraryX) with significantly different statistical properties. The optimal method changes depending on the error considered and the characteristics of the synthetic time series in the time window that is being processed. Figure 4 considers *MaAE*, Figure 5 *MeAE*, Figure 6 *NRMS* and Figure 7 *PRD*. The figures show the original time series values (blue), the recovered time series values (orange) and the error in each point (green). The analysis window size is fixed to 200 points and from each window the maximum number of points that can be selected ($k$) is fixed to 50. In each window, the optimal method is marked with a green background.

In all cases, the method that is selected as optimal in most of the windows is the *lttb* algorithm. This effect is more notable where a mean or cumulative value of point to point errors is used. By contrast, if the importance to extreme values is given, for example using maximum value, when a function such as *MaAE* is used, the optimal method depends more on the local characteristics of the window studied. Thus, Figure 4 shows that it is not a clear winner when it comes

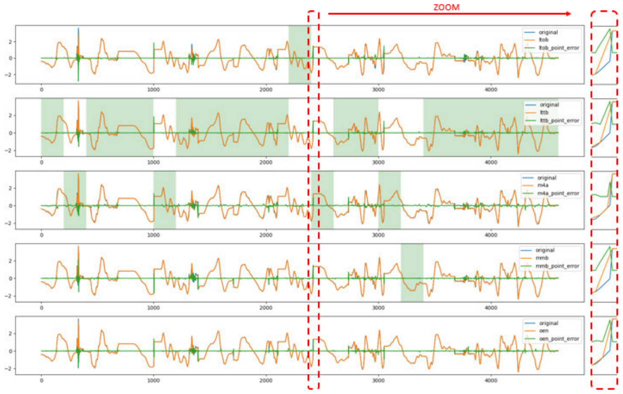


**FIGURE 6.** $y_o$ (in blue), $y_r$ (in orange), $y_r - y_o$ (in green) time series from a synthetically generated time series. The optimal method selected by *NRMS* in each window is highlighted. A detailed view of the some representative time segment is added. At right, a detailed view of a representative time segment is added.

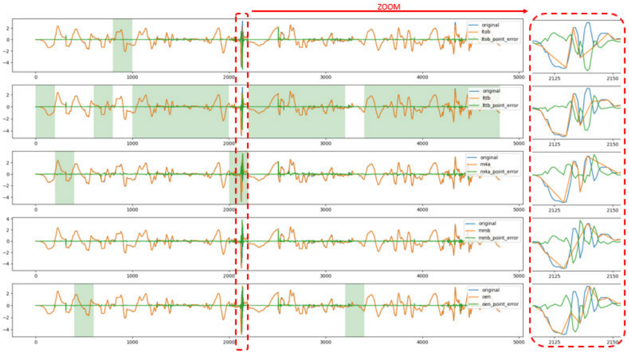


**FIGURE 7.** $y_o$ (in blue), $y_r$ (in orange), $y_r - y_o$ (in green) time series from a synthetically generated time series. The optimal method selected by *PRD* in each window is highlighted. At right, a detailed view of a representative time segment is added.

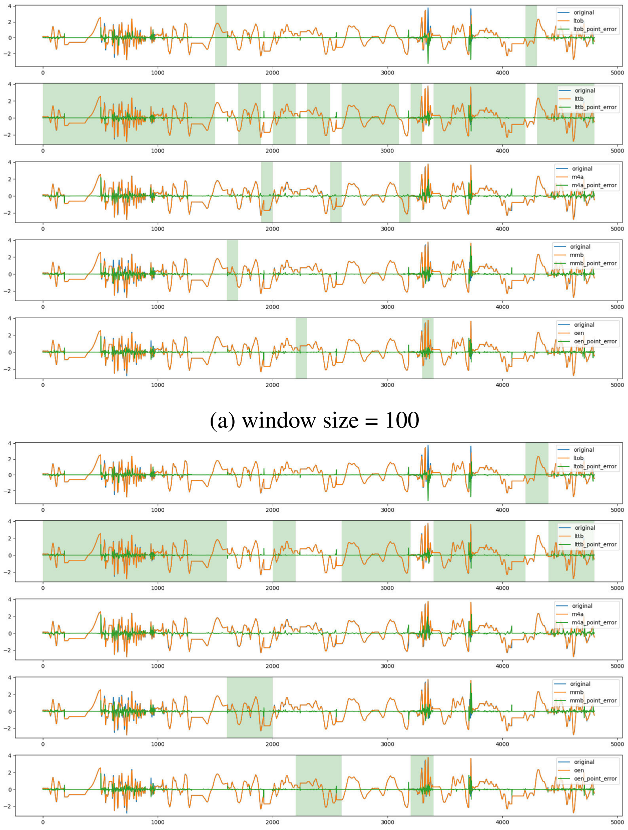
to point selection strategies, as all depends on the time series characteristics and on the error used to measure it.

In Figure 8 the effect of the window size selection is shown. Even if the total number of points in the complete time series is the same, the optimal method distribution changes. In a smaller window size, the quantity of points to select from is reduced so the algorithm can select them quicker. Furthermore, a smaller window allows adjusting the algorithm to the actual time series characteristics. However, a bigger window size could help distributing the points in time in an smarter way.

It is important, therefore, that experiments are made with data from different origins and characteristics in order to ensure that the selection of the algorithm does not only depend on the error function used. For that purpose, an extensive experimentation is needed and this is shown in sections IV and V.

## IV. EXPERIMENTAL SETUP

The experiments presented in this Section use the UCR Time Series data archive introduced in Section III-B is used.



(a) window size = 100

(b) window size = 200

**FIGURE 8.** Effect of window size parameter in the optimal method selection.

In all experiments, each dataset from the Archive is studied separately. Furthermore, from each dataset, each time series contained in the train or test set is used as an independent time series for point selection algorithm applications.

There are two possible strategies to define window size or batch length:

1) Based on a temporal window to schedule the data points selection periodically
2) Based on memory limitations that raise the data points selection algorithm when a reduction is needed.

In the particular case of having equidistant points, both previous cases arrive into the same definition of window size or batch length.

Time series in UCR Time Series Classification Archive do not have a time index. Therefore, an uniformly sampled index is used as time index of the series. As an uniformed time sampling is used, both strategies detailed above are equivalent. For each dataset, each time series is taken independently and the objective is to reduce at least 50% of the data points available in each of the time series. If the length of time series is variant among the dataset, the maximum length of the time series is taken to define the value of $k$ in the downsampling

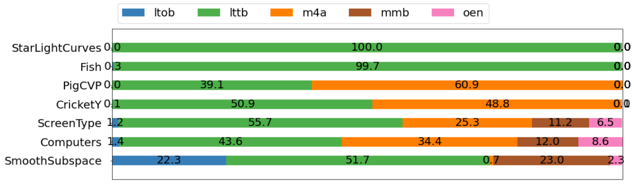


**FIGURE 9.** **Zoom of the Figures 13 and 14 that are commented in the discussion for an easier comparison.**

methodologies. Hence, $k$ value corresponds to:

$$k = truncate\left(\frac{max(length(y_o))}{2}\right) \tag{6}$$

This $k$ value was selected in order to ensure that in the inner buckets generated by the algorithms *lttb* and *ltob* contain at least two points. For each time series available in the dataset the fit method is applied, i.e, all available point selection algorithms are tried for compressing the signal and the optimal solution is saved.

## V. EXPERIMENTAL RESULTS

Tables and Figures with the results of all datasets from UCR Time Series Classification Archive can be found in A.

For each time series from each dataset, the mean error value between $y_r$ and $y_o$ is used. Tables 2 and 3 report the mean and standard deviation values of all the errors among datasets for each method. The *opt* column presents the mean and standard deviation values in the case of using the optimal method (minimal error) in each of the time series. The datasets are sorted in ascending order starting with the dataset with the minimal mean value of the mean of errors of all the time series contained in the dataset. Similar information is shown visually in Figure 12. Due to printable table dimension limitations, datasets have been grouped in 8 different groups (16 datasets per group) in the same order that appear in Tables 2 and 3 and the sum of mean errors per method are shown in the Table 1.

**TABLE 1.** **Grouped sum of mean errors of *MeAE* values obtained from each time series for datasets from URC Time Series Classification Archive.**

| datasets group | opt | ltob | lttb | m4a | mmb | oen |
|---|---|---|---|---|---|---|
| group 1 | **0.377** | 0.493 | 0.386 | 0.695 | 0.635 | 0.657 |
| group 2 | **0.853** | 1.22 | 0.863 | 1.585 | 1.334 | 1.29 |
| group 3 | **1.508** | 2.026 | 1.526 | 2.79 | 2.243 | 2.203 |
| group 4 | **2.74** | 3.821 | 2.892 | 4.15 | 4.192 | 4.052 |
| group 5 | **4.493** | 6.016 | 4.674 | 6.506 | 6.956 | 7.134 |
| group 6 | **7.055** | 9.632 | 7.306 | 10.278 | 10.996 | 10.795 |
| group 7 | **10.384** | 14.267 | 10.698 | 17.227 | 17.429 | 17.093 |
| group 8 | **59.529** | 89.281 | 66.442 | 75.056 | 75.641 | 87.967 |
| **total sum** | **86.939** | **126.756** | **94.787** | **118.287** | **119.426** | **131.191** |

In general, the *lttb* method when using the *MeAE* is the most adequate method when different datasets are grouped. Moreover, selecting the optimal method in each time window when the difference between $y_r$ and $y_o$ time series is greater becomes more important. In other words, when the selecting points are not enough to preserve all the data adequately, selecting the optimal points each time could have a greater impact. This is shown in Table 1 as the difference between choosing the optimal method in each window each time (column *opt* of the table) or using the same method for all the dataset (rest of the columns). The total sum of the grouped *MeAE* using the optimal point selection method in each time series is 86.939 that has nearly eight point difference compared to globally optimal methodology *lttb* value 94.787. Furthermore, this difference becomes much bigger when if another algorithm from the set $D$ apart from *lttb* is selected, with value ranges from 31 to 44 points.

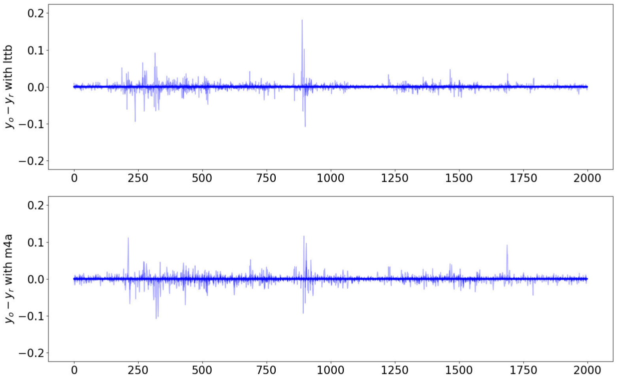


**FIGURE 10.** **Difference between $y_o$ and $y_r$ when different point selection algorithms from $D$ are applied to the PigCVP dataset.**

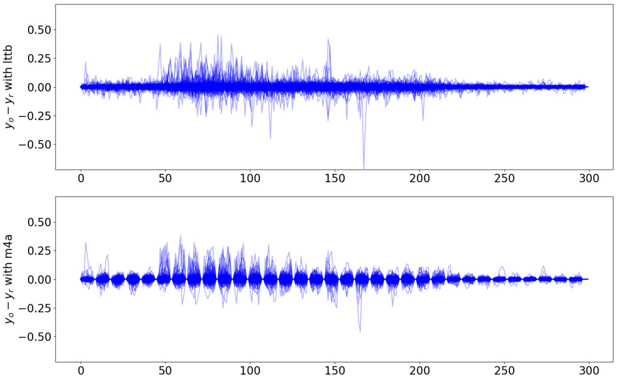


**FIGURE 11.** **Difference between $y_o$ and $y_r$ when different point selection algorithms from $D$ are applied to the CricketY dataset.**

Each downsampling method has its own characteristics and adaptability depending on the time series properties. In case of having a variable that has a static or similar properties during all the time range, it is possible to select an unique optimal downsampling method that suits adequately the needs. However, this is not the usual case in real sensor data as process properties may vary with time and there is an stochastic part of the variable that cannot be controlled. In Figures 13 and 14 for each dataset, the percentage of the number of times each method has become the optimal one for downsampling a time

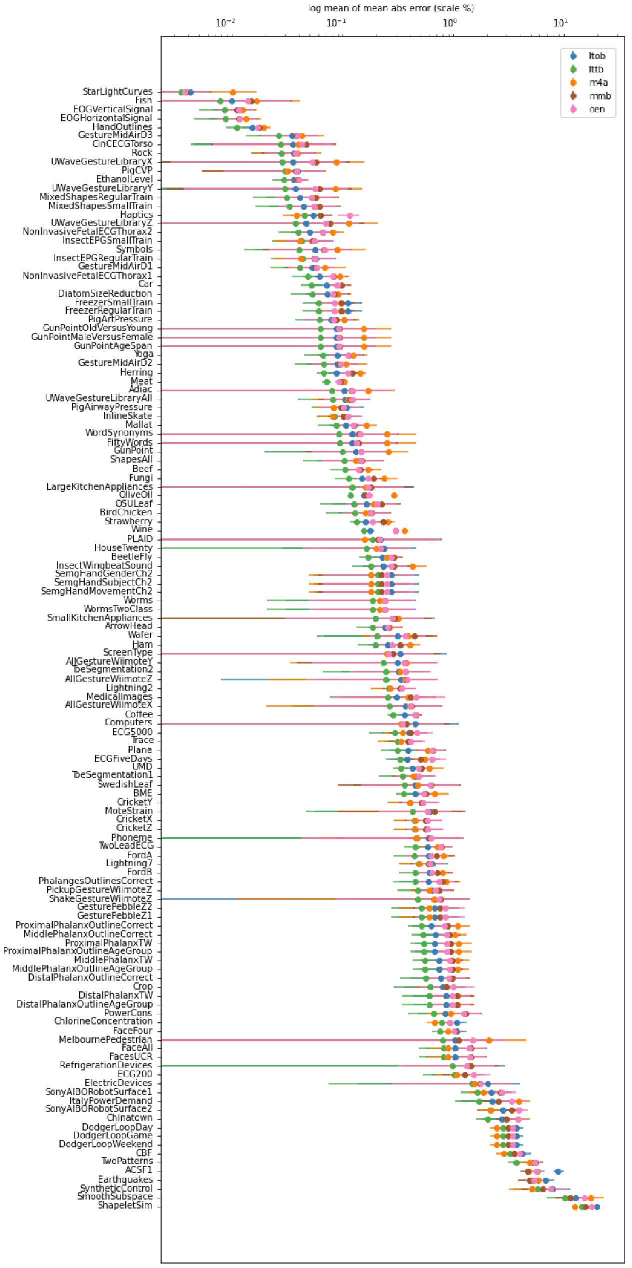


**FIGURE 12.** Mean value per dataset of *MeAE* values obtained for each time series of the dataset.

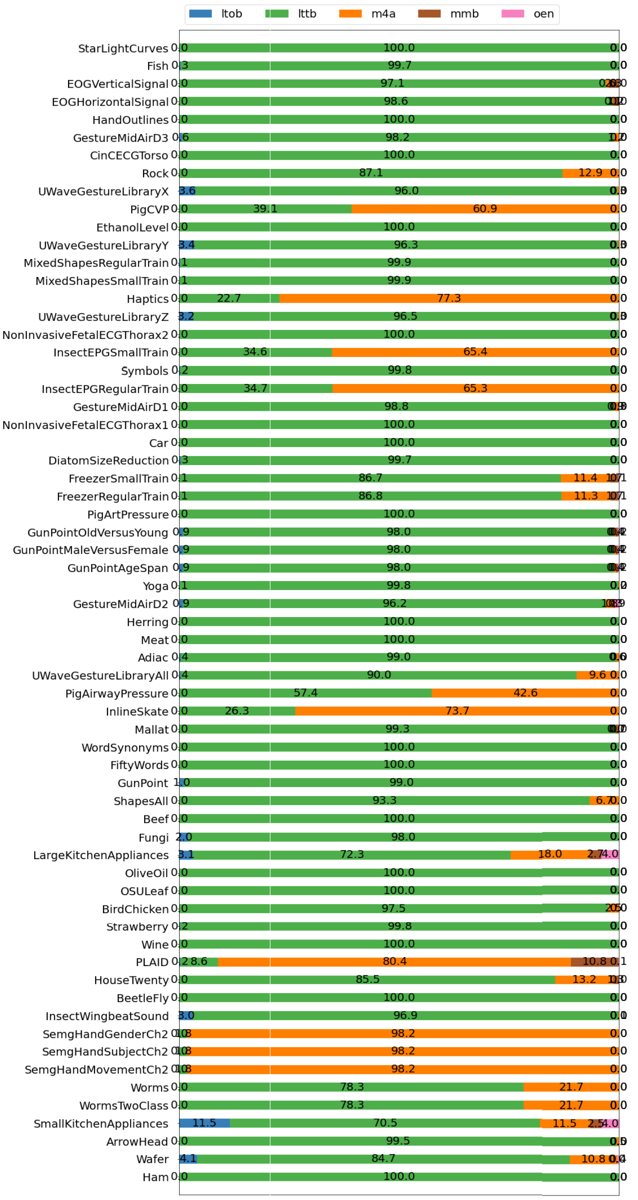


**FIGURE 13.** Distribution by dataset of the optimal data points selection method as determined by the use of *MeAE* per time series separately for datasets in Subset 1 of the UCR archive.

series (an instance of the dataset, equivalent to a specific time window) is shown.

In some cases, a unique method is responsible of being the most adequate in more than 90% of the cases, as it happens in StarlightCurves and Fish datasets. However, the selection for an adequate downsampling method is not obvious for others, such as for the PigCVP and CricketY datasets where the optimal solution is divided between *lltb* and *m4a* methods or even more distributed as it happens for datasets ScreenType, Computers and SmoothSubspace. In Figure 9 a zoom of Figures 13 and 14 is shown for datasets mentioned in both cases.

Figures 10 and 11 show point to point errors between $y_o$ and $y_r$ time series of the PigCVP and CricketY datasets where the optimal point selection strategy is not obvious. In both Figures, certain peaks of errors that appear using one of the methods are considerably reduced by the use of the other method.

This experiment shows that a methodology to adaptively select or / and monitor the point selection strategy is needed. Not all the time windows of a certain time series can be treated equally and even less when it comes to a real sensor data where all the context variables are not totally controlled.

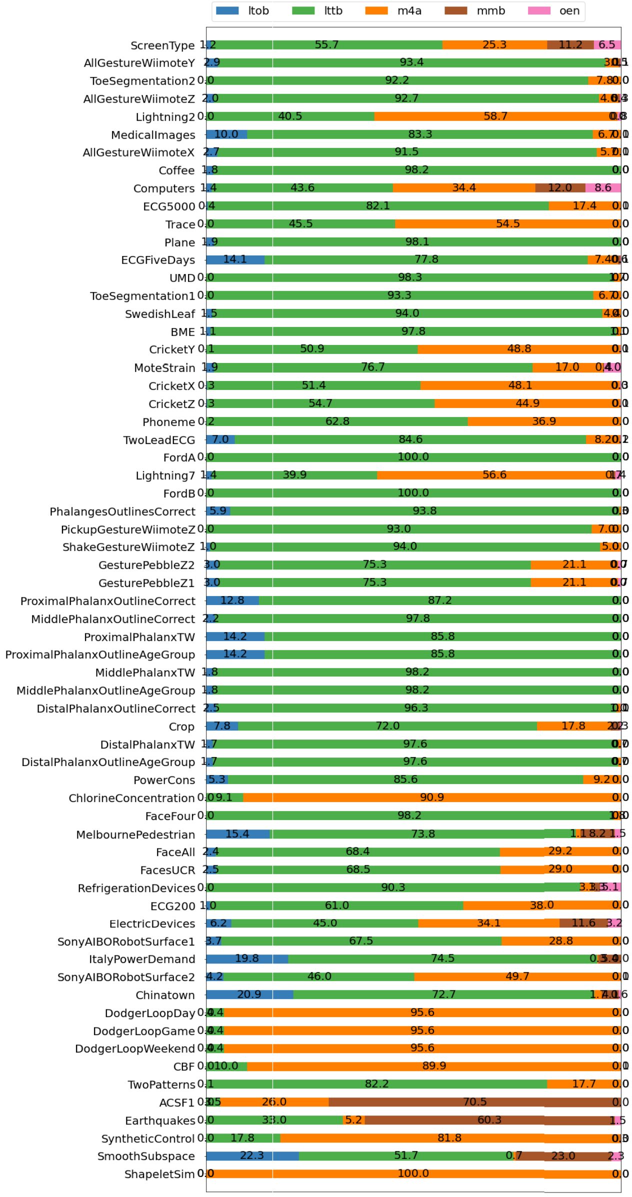


**FIGURE 14.** Distribution by dataset of the optimal data points selection method as determined by the use of *MeAE* per time series separately for datasets in Subset 2 of the UCR archive.

A change in the process, the dependency on other internal or external variables or even noise can affect the point selection method. Controlling the error values is critical in order to detect a point selection model drift and, in consequence, retrain the point selection strategy to maintain point selection quality.

## VI. CONCLUSION AND FUTURE WORK

Considering the need to adaptively compress time series from stochastic processes into streams with constant or limited length in order to meet memory capacity limitations, this paper has put forward and demonstrated in a practical implementation the idea of combining different data points selection methodologies using their potential in the current signal window, while providing a definition of errors for the determination of the optimal data points selection methodology in each moment, and for different characteristics of interest relative depending on the envisaged application.

The proposed methods have been implemented and applied to a wide variety of real-world time series datasets from a public open database, demonstrating their value for the characterization and the compression of the data.

Future work to be considered includes combining algorithms to select points using a maximum value error for local errors, the ones that appeared in section II-A, in windows where maximum memory limitation per window is satisfied with methodologies that use maximum number of segments. With these combinations, it is possible to work with a trade off between maximum error allowed and memory size control. Furthermore, it should be possible to select points for multiple time series at once, as all of them will be saved in the same dataset or need to be synchronized, for example to be able to plot them in multiple dimensions. Finally, controlling the window size and quantity of data points to be selected depending on the characteristics of the time series would prove beneficial in a number of application scenarios [28]. Jain *et al.* [16] introduce an adaptive resampling frequency depending on the time series characteristics. The idea is using the actual error values not only to retrain the point selection strategy, but also to select adequate input parameters.

**TABLE 2.** Per-dataset mean and standard deviation values of the *MeAE* values obtained for each time series and by each method separately for Subset 1 of the UCR archive. The optimal column shows the mean and standard deviation values of the *MeAE* values when the optimal point selection method is selected for each time series of the dataset.

| dataset | opt | ltob | lttb | m4a | mmb | oen |
|---|---|---|---|---|---|---|
| StarLightCurves | $0.004 \pm 0.002$ | $0.004 \pm 0.002$ | $0.004 \pm 0.002$ | $0.01 \pm 0.006$ | $0.004 \pm 0.002$ | $0.004 \pm 0.002$ |
| Fish | $0.008 \pm 0.01$ | $0.01 \pm 0.01$ | $0.008 \pm 0.01$ | $0.017 \pm 0.024$ | $0.015 \pm 0.019$ | $0.014 \pm 0.019$ |
| EOGVerticalSignal | $0.009 \pm 0.004$ | $0.011 \pm 0.005$ | $0.009 \pm 0.004$ | $0.012 \pm 0.004$ | $0.011 \pm 0.004$ | $0.011 \pm 0.004$ |
| EOGHorizontalSignal | $0.009 \pm 0.004$ | $0.012 \pm 0.006$ | $0.009 \pm 0.004$ | $0.013 \pm 0.005$ | $0.012 \pm 0.004$ | $0.012 \pm 0.004$ |
| HandOutlines | $0.011 \pm 0.002$ | $0.015 \pm 0.003$ | $0.011 \pm 0.002$ | $0.019 \pm 0.003$ | $0.017 \pm 0.003$ | $0.017 \pm 0.003$ |
| GestureMidAirD3 | $0.026 \pm 0.013$ | $0.035 \pm 0.018$ | $0.027 \pm 0.014$ | $0.043 \pm 0.024$ | $0.038 \pm 0.019$ | $0.039 \pm 0.02$ |
| CinCECGTorso | $0.027 \pm 0.023$ | $0.035 \pm 0.031$ | $0.027 \pm 0.023$ | $0.039 \pm 0.027$ | $0.046 \pm 0.04$ | $0.042 \pm 0.036$ |
| Rock | $0.027 \pm 0.013$ | $0.036 \pm 0.018$ | $0.028 \pm 0.013$ | $0.039 \pm 0.024$ | $0.037 \pm 0.019$ | $0.037 \pm 0.018$ |
| UWaveGestureLibraryX | $0.029 \pm 0.03$ | $0.036 \pm 0.038$ | $0.029 \pm 0.03$ | $0.087 \pm 0.068$ | $0.058 \pm 0.057$ | $0.053 \pm 0.051$ |
| PigCVP | $0.029 \pm 0.019$ | $0.039 \pm 0.024$ | $0.03 \pm 0.019$ | $0.031 \pm 0.026$ | $0.038 \pm 0.032$ | $0.038 \pm 0.03$ |
| EthanolLevel | $0.029 \pm 0.006$ | $0.036 \pm 0.008$ | $0.029 \pm 0.006$ | $0.039 \pm 0.007$ | $0.04 \pm 0.008$ | $0.04 \pm 0.008$ |
| UWaveGestureLibraryY | $0.03 \pm 0.032$ | $0.038 \pm 0.041$ | $0.03 \pm 0.032$ | $0.085 \pm 0.062$ | $0.061 \pm 0.059$ | $0.056 \pm 0.053$ |
| MixedShapesRegularTrain | $0.031 \pm 0.016$ | $0.041 \pm 0.021$ | $0.031 \pm 0.016$ | $0.053 \pm 0.023$ | $0.058 \pm 0.032$ | $0.052 \pm 0.028$ |
| MixedShapesSmallTrain | $0.033 \pm 0.017$ | $0.044 \pm 0.023$ | $0.033 \pm 0.017$ | $0.056 \pm 0.025$ | $0.061 \pm 0.035$ | $0.056 \pm 0.03$ |
| Haptics | $0.037 \pm 0.009$ | $0.054 \pm 0.013$ | $0.045 \pm 0.011$ | $0.038 \pm 0.01$ | $0.063 \pm 0.016$ | $0.114 \pm 0.025$ |
| UWaveGestureLibraryZ | $0.037 \pm 0.039$ | $0.046 \pm 0.049$ | $0.038 \pm 0.039$ | $0.113 \pm 0.092$ | $0.076 \pm 0.077$ | $0.071 \pm 0.068$ |
| NonInvasiveFetalECGThorax2 | $0.04 \pm 0.014$ | $0.051 \pm 0.017$ | $0.04 \pm 0.014$ | $0.082 \pm 0.019$ | $0.067 \pm 0.026$ | $0.067 \pm 0.025$ |
| InsectEPGSmallTrain | $0.04 \pm 0.017$ | $0.055 \pm 0.026$ | $0.042 \pm 0.02$ | $0.041 \pm 0.017$ | $0.053 \pm 0.023$ | $0.055 \pm 0.025$ |
| Symbols | $0.04 \pm 0.028$ | $0.056 \pm 0.039$ | $0.04 \pm 0.028$ | $0.089 \pm 0.069$ | $0.068 \pm 0.049$ | $0.068 \pm 0.047$ |
| InsectEPGRegularTrain | $0.041 \pm 0.019$ | $0.057 \pm 0.028$ | $0.044 \pm 0.022$ | $0.042 \pm 0.018$ | $0.055 \pm 0.024$ | $0.056 \pm 0.027$ |
| GestureMidAirD1 | $0.041 \pm 0.019$ | $0.053 \pm 0.026$ | $0.041 \pm 0.019$ | $0.069 \pm 0.035$ | $0.057 \pm 0.026$ | $0.058 \pm 0.026$ |
| NonInvasiveFetalECGThorax1 | $0.048 \pm 0.015$ | $0.062 \pm 0.019$ | $0.048 \pm 0.015$ | $0.094 \pm 0.019$ | $0.082 \pm 0.027$ | $0.081 \pm 0.023$ |
| Car | $0.052 \pm 0.011$ | $0.071 \pm 0.015$ | $0.052 \pm 0.011$ | $0.089 \pm 0.017$ | $0.098 \pm 0.019$ | $0.089 \pm 0.017$ |
| DiatomSizeReduction | $0.053 \pm 0.019$ | $0.072 \pm 0.027$ | $0.053 \pm 0.019$ | $0.091 \pm 0.026$ | $0.084 \pm 0.03$ | $0.081 \pm 0.028$ |
| FreezerSmallTrain | $0.058 \pm 0.016$ | $0.111 \pm 0.036$ | $0.06 \pm 0.018$ | $0.095 \pm 0.032$ | $0.098 \pm 0.027$ | $0.084 \pm 0.02$ |
| FreezerRegularTrain | $0.058 \pm 0.016$ | $0.111 \pm 0.036$ | $0.06 \pm 0.018$ | $0.095 \pm 0.032$ | $0.098 \pm 0.027$ | $0.084 \pm 0.02$ |
| PigArtPressure | $0.061 \pm 0.025$ | $0.078 \pm 0.031$ | $0.061 \pm 0.025$ | $0.103 \pm 0.035$ | $0.087 \pm 0.042$ | $0.083 \pm 0.039$ |
| GunPointOldVersusYoung | $0.062 \pm 0.069$ | $0.088 \pm 0.111$ | $0.062 \pm 0.07$ | $0.154 \pm 0.114$ | $0.093 \pm 0.102$ | $0.092 \pm 0.104$ |
| GunPointMaleVersusFemale | $0.062 \pm 0.069$ | $0.088 \pm 0.111$ | $0.062 \pm 0.07$ | $0.154 \pm 0.114$ | $0.093 \pm 0.102$ | $0.092 \pm 0.104$ |
| GunPointAgeSpan | $0.062 \pm 0.069$ | $0.088 \pm 0.111$ | $0.062 \pm 0.07$ | $0.154 \pm 0.114$ | $0.093 \pm 0.102$ | $0.092 \pm 0.104$ |
| Yoga | $0.066 \pm 0.022$ | $0.089 \pm 0.031$ | $0.066 \pm 0.022$ | $0.124 \pm 0.039$ | $0.114 \pm 0.038$ | $0.112 \pm 0.037$ |
| GestureMidAirD2 | $0.067 \pm 0.031$ | $0.09 \pm 0.043$ | $0.068 \pm 0.032$ | $0.107 \pm 0.056$ | $0.094 \pm 0.043$ | $0.095 \pm 0.044$ |
| Herring | $0.068 \pm 0.01$ | $0.088 \pm 0.015$ | $0.068 \pm 0.01$ | $0.144 \pm 0.018$ | $0.121 \pm 0.017$ | $0.111 \pm 0.017$ |
| Meat | $0.071 \pm 0.007$ | $0.097 \pm 0.011$ | $0.071 \pm 0.007$ | $0.104 \pm 0.005$ | $0.095 \pm 0.008$ | $0.092 \pm 0.007$ |
| Adiac | $0.079 \pm 0.115$ | $0.104 \pm 0.16$ | $0.08 \pm 0.116$ | $0.168 \pm 0.121$ | $0.119 \pm 0.152$ | $0.121 \pm 0.149$ |
| UWaveGestureLibraryAll | $0.08 \pm 0.043$ | $0.105 \pm 0.053$ | $0.081 \pm 0.042$ | $0.111 \pm 0.06$ | $0.117 \pm 0.057$ | $0.121 \pm 0.054$ |
| PigAirwayPressure | $0.081 \pm 0.029$ | $0.108 \pm 0.043$ | $0.083 \pm 0.033$ | $0.083 \pm 0.028$ | $0.096 \pm 0.034$ | $0.102 \pm 0.039$ |
| InlineSkate | $0.081 \pm 0.024$ | $0.11 \pm 0.036$ | $0.084 \pm 0.027$ | $0.081 \pm 0.024$ | $0.102 \pm 0.029$ | $0.112 \pm 0.033$ |
| Mallat | $0.088 \pm 0.028$ | $0.108 \pm 0.033$ | $0.088 \pm 0.028$ | $0.162 \pm 0.038$ | $0.127 \pm 0.039$ | $0.123 \pm 0.039$ |
| WordSynonyms | $0.093 \pm 0.098$ | $0.122 \pm 0.135$ | $0.093 \pm 0.098$ | $0.252 \pm 0.202$ | $0.139 \pm 0.178$ | $0.143 \pm 0.165$ |
| FiftyWords | $0.094 \pm 0.099$ | $0.123 \pm 0.134$ | $0.094 \pm 0.099$ | $0.252 \pm 0.202$ | $0.138 \pm 0.173$ | $0.134 \pm 0.162$ |
| GunPoint | $0.099 \pm 0.076$ | $0.13 \pm 0.111$ | $0.099 \pm 0.076$ | $0.257 \pm 0.123$ | $0.145 \pm 0.1$ | $0.147 \pm 0.094$ |
| ShapesAll | $0.104 \pm 0.06$ | $0.144 \pm 0.087$ | $0.104 \pm 0.06$ | $0.131 \pm 0.063$ | $0.148 \pm 0.082$ | $0.146 \pm 0.082$ |
| Beef | $0.105 \pm 0.029$ | $0.142 \pm 0.04$ | $0.105 \pm 0.029$ | $0.171 \pm 0.047$ | $0.144 \pm 0.041$ | $0.138 \pm 0.039$ |
| Fungi | $0.112 \pm 0.03$ | $0.15 \pm 0.044$ | $0.112 \pm 0.03$ | $0.236 \pm 0.069$ | $0.188 \pm 0.055$ | $0.169 \pm 0.039$ |
| LargeKitchenAppliances | $0.113 \pm 0.138$ | $0.179 \pm 0.258$ | $0.123 \pm 0.164$ | $0.161 \pm 0.149$ | $0.18 \pm 0.24$ | $0.17 \pm 0.183$ |
| OliveOil | $0.116 \pm 0.001$ | $0.154 \pm 0.004$ | $0.116 \pm 0.001$ | $0.287 \pm 0.003$ | $0.16 \pm 0.002$ | $0.173 \pm 0.001$ |
| OSULeaf | $0.126 \pm 0.065$ | $0.164 \pm 0.086$ | $0.126 \pm 0.065$ | $0.19 \pm 0.071$ | $0.224 \pm 0.107$ | $0.202 \pm 0.097$ |
| BirdChicken | $0.127 \pm 0.057$ | $0.176 \pm 0.087$ | $0.128 \pm 0.057$ | $0.16 \pm 0.059$ | $0.184 \pm 0.083$ | $0.18 \pm 0.082$ |
| Strawberry | $0.134 \pm 0.018$ | $0.159 \pm 0.022$ | $0.134 \pm 0.018$ | $0.255 \pm 0.034$ | $0.231 \pm 0.042$ | $0.187 \pm 0.022$ |
| Wine | $0.155 \pm 0.006$ | $0.177 \pm 0.011$ | $0.155 \pm 0.006$ | $0.36 \pm 0.013$ | $0.301 \pm 0.011$ | $0.3 \pm 0.013$ |
| PLAID | $0.155 \pm 0.386$ | $0.22 \pm 0.552$ | $0.186 \pm 0.46$ | $0.158 \pm 0.391$ | $0.209 \pm 0.522$ | $0.214 \pm 0.535$ |
| HouseTwenty | $0.158 \pm 0.151$ | $0.238 \pm 0.211$ | $0.162 \pm 0.161$ | $0.199 \pm 0.15$ | $0.218 \pm 0.156$ | $0.221 \pm 0.178$ |
| BeetleFly | $0.171 \pm 0.031$ | $0.226 \pm 0.044$ | $0.171 \pm 0.031$ | $0.25 \pm 0.043$ | $0.29 \pm 0.05$ | $0.273 \pm 0.043$ |
| InsectWingbeatSound | $0.179 \pm 0.06$ | $0.23 \pm 0.081$ | $0.179 \pm 0.06$ | $0.423 \pm 0.138$ | $0.29 \pm 0.112$ | $0.271 \pm 0.102$ |
| SemgHandGenderCh2 | $0.18 \pm 0.132$ | $0.273 \pm 0.201$ | $0.207 \pm 0.153$ | $0.18 \pm 0.132$ | $0.222 \pm 0.164$ | $0.249 \pm 0.184$ |
| SemgHandSubjectCh2 | $0.18 \pm 0.132$ | $0.273 \pm 0.201$ | $0.207 \pm 0.153$ | $0.18 \pm 0.132$ | $0.222 \pm 0.164$ | $0.249 \pm 0.184$ |
| SemgHandMovementCh2 | $0.18 \pm 0.132$ | $0.273 \pm 0.201$ | $0.207 \pm 0.153$ | $0.18 \pm 0.132$ | $0.222 \pm 0.164$ | $0.249 \pm 0.184$ |
| Worms | $0.181 \pm 0.15$ | $0.24 \pm 0.21$ | $0.185 \pm 0.165$ | $0.215 \pm 0.153$ | $0.239 \pm 0.181$ | $0.241 \pm 0.192$ |
| WormsTwoClass | $0.181 \pm 0.15$ | $0.24 \pm 0.21$ | $0.185 \pm 0.165$ | $0.215 \pm 0.153$ | $0.239 \pm 0.181$ | $0.241 \pm 0.192$ |
| SmallKitchenAppliances | $0.183 \pm 0.197$ | $0.293 \pm 0.367$ | $0.198 \pm 0.254$ | $0.318 \pm 0.263$ | $0.295 \pm 0.333$ | $0.28 \pm 0.25$ |
| ArrowHead | $0.186 \pm 0.055$ | $0.241 \pm 0.073$ | $0.186 \pm 0.055$ | $0.262 \pm 0.078$ | $0.261 \pm 0.082$ | $0.253 \pm 0.075$ |
| Wafer | $0.194 \pm 0.14$ | $0.309 \pm 0.243$ | $0.204 \pm 0.147$ | $0.391 \pm 0.243$ | $0.443 \pm 0.255$ | $0.365 \pm 0.192$ |
| Ham | $0.197 \pm 0.063$ | $0.252 \pm 0.086$ | $0.197 \pm 0.063$ | $0.406 \pm 0.087$ | $0.327 \pm 0.111$ | $0.278 \pm 0.093$ |

## APPENDIX
## COMPLETE RESULTS OF UCR ARCHIVE

See Tables 2 and 3.

**TABLE 3. Per-dataset mean and standard deviation values of the *MeAE* values obtained for each time series and by each method separately for Subset 2 of the UCR archive. The optimal column shows the mean and standard deviation values of the *MeAE* values when the optimal point selection method is selected for each time series of the dataset.**

| dataset | opt | ltob | lttb | m4a | mmb | oen |
|---|---|---|---|---|---|---|
| ScreenType | 0.21 ± 0.309 | 0.332 ± 0.52 | 0.26 ± 0.423 | 0.251 ± 0.337 | 0.287 ± 0.472 | 0.26 ± 0.387 |
| AllGestureWiimoteY | 0.229 ± 0.189 | 0.31 ± 0.261 | 0.231 ± 0.191 | 0.376 ± 0.343 | 0.363 ± 0.325 | 0.36 ± 0.309 |
| ToeSegmentation2 | 0.24 ± 0.162 | 0.318 ± 0.221 | 0.245 ± 0.179 | 0.332 ± 0.157 | 0.365 ± 0.232 | 0.364 ± 0.252 |
| AllGestureWiimoteZ | 0.242 ± 0.211 | 0.334 ± 0.326 | 0.244 ± 0.213 | 0.363 ± 0.342 | 0.376 ± 0.321 | 0.379 ± 0.334 |
| Lightning2 | 0.244 ± 0.076 | 0.344 ± 0.11 | 0.261 ± 0.083 | 0.268 ± 0.093 | 0.33 ± 0.105 | 0.344 ± 0.108 |
| MedicalImages | 0.248 ± 0.162 | 0.307 ± 0.21 | 0.256 ± 0.181 | 0.391 ± 0.171 | 0.41 ± 0.271 | 0.456 ± 0.377 |
| AllGestureWiimoteX | 0.262 ± 0.227 | 0.364 ± 0.326 | 0.265 ± 0.231 | 0.402 ± 0.382 | 0.416 ± 0.355 | 0.414 ± 0.359 |
| Coffee | 0.283 ± 0.034 | 0.362 ± 0.05 | 0.283 ± 0.035 | 0.452 ± 0.044 | 0.461 ± 0.052 | 0.452 ± 0.055 |
| Computers | 0.286 ± 0.373 | 0.45 ± 0.646 | 0.341 ± 0.49 | 0.338 ± 0.431 | 0.372 ± 0.538 | 0.349 ± 0.465 |
| ECG5000 | 0.288 ± 0.111 | 0.392 ± 0.172 | 0.294 ± 0.125 | 0.345 ± 0.118 | 0.413 ± 0.155 | 0.471 ± 0.171 |
| Trace | 0.295 ± 0.08 | 0.403 ± 0.107 | 0.311 ± 0.079 | 0.336 ± 0.133 | 0.389 ± 0.135 | 0.413 ± 0.126 |
| Plane | 0.311 ± 0.091 | 0.386 ± 0.122 | 0.311 ± 0.091 | 0.582 ± 0.144 | 0.647 ± 0.19 | 0.637 ± 0.169 |
| ECGFiveDays | 0.317 ± 0.078 | 0.379 ± 0.118 | 0.329 ± 0.091 | 0.553 ± 0.184 | 0.507 ± 0.123 | 0.644 ± 0.204 |
| UMD | 0.334 ± 0.056 | 0.428 ± 0.088 | 0.334 ± 0.056 | 0.604 ± 0.198 | 0.512 ± 0.112 | 0.489 ± 0.093 |
| ToeSegmentation1 | 0.348 ± 0.139 | 0.454 ± 0.183 | 0.35 ± 0.14 | 0.433 ± 0.146 | 0.487 ± 0.18 | 0.485 ± 0.188 |
| SwedishLeaf | 0.356 ± 0.232 | 0.453 ± 0.293 | 0.358 ± 0.236 | 0.478 ± 0.27 | 0.622 ± 0.533 | 0.615 ± 0.472 |
| BME | 0.356 ± 0.064 | 0.448 ± 0.09 | 0.356 ± 0.063 | 0.673 ± 0.226 | 0.566 ± 0.136 | 0.539 ± 0.11 |
| CricketY | 0.384 ± 0.141 | 0.52 ± 0.198 | 0.405 ± 0.157 | 0.406 ± 0.146 | 0.516 ± 0.196 | 0.536 ± 0.201 |
| MoteStrain | 0.388 ± 0.334 | 0.621 ± 0.566 | 0.428 ± 0.382 | 0.587 ± 0.498 | 0.676 ± 0.587 | 0.578 ± 0.369 |
| CricketX | 0.419 ± 0.142 | 0.567 ± 0.202 | 0.441 ± 0.155 | 0.449 ± 0.166 | 0.557 ± 0.199 | 0.583 ± 0.205 |
| CricketZ | 0.422 ± 0.145 | 0.572 ± 0.206 | 0.444 ± 0.159 | 0.454 ± 0.169 | 0.562 ± 0.202 | 0.586 ± 0.208 |
| Phoneme | 0.431 ± 0.418 | 0.592 ± 0.62 | 0.468 ± 0.486 | 0.462 ± 0.403 | 0.605 ± 0.517 | 0.63 ± 0.589 |
| TwoLeadECG | 0.441 ± 0.093 | 0.584 ± 0.167 | 0.449 ± 0.097 | 0.713 ± 0.206 | 0.781 ± 0.17 | 0.766 ± 0.169 |
| FordA | 0.445 ± 0.159 | 0.594 ± 0.204 | 0.445 ± 0.159 | 0.818 ± 0.147 | 0.707 ± 0.306 | 0.628 ± 0.257 |
| Lightning7 | 0.446 ± 0.152 | 0.637 ± 0.235 | 0.488 ± 0.169 | 0.496 ± 0.181 | 0.595 ± 0.2 | 0.62 ± 0.202 |
| FordB | 0.447 ± 0.132 | 0.6 ± 0.166 | 0.447 ± 0.132 | 0.794 ± 0.115 | 0.718 ± 0.256 | 0.636 ± 0.215 |
| PhalangesOutlinesCorrect | 0.451 ± 0.175 | 0.597 ± 0.279 | 0.453 ± 0.174 | 0.865 ± 0.26 | 0.795 ± 0.299 | 0.741 ± 0.271 |
| PickupGestureWiimoteZ | 0.472 ± 0.165 | 0.642 ± 0.228 | 0.475 ± 0.169 | 0.607 ± 0.205 | 0.742 ± 0.257 | 0.686 ± 0.222 |
| ShakeGestureWiimoteZ | 0.475 ± 0.433 | 0.679 ± 0.718 | 0.478 ± 0.437 | 0.668 ± 0.658 | 0.765 ± 0.629 | 0.741 ± 0.657 |
| GesturePebbleZ2 | 0.486 ± 0.215 | 0.673 ± 0.356 | 0.509 ± 0.239 | 0.602 ± 0.279 | 0.759 ± 0.344 | 0.847 ± 0.401 |
| GesturePebbleZ1 | 0.486 ± 0.215 | 0.673 ± 0.356 | 0.509 ± 0.239 | 0.602 ± 0.279 | 0.759 ± 0.344 | 0.847 ± 0.401 |
| ProximalPhalanxOutlineCorrect | 0.505 ± 0.133 | 0.633 ± 0.216 | 0.51 ± 0.131 | 1.084 ± 0.301 | 0.895 ± 0.222 | 0.831 ± 0.177 |
| MiddlePhalanxOutlineCorrect | 0.526 ± 0.119 | 0.689 ± 0.205 | 0.527 ± 0.119 | 1.041 ± 0.238 | 0.926 ± 0.214 | 0.881 ± 0.217 |
| ProximalPhalanxTW | 0.531 ± 0.139 | 0.674 ± 0.234 | 0.537 ± 0.134 | 1.12 ± 0.315 | 0.945 ± 0.234 | 0.87 ± 0.176 |
| ProximalPhalanxOutlineAgeGroup | 0.531 ± 0.139 | 0.674 ± 0.234 | 0.537 ± 0.134 | 1.12 ± 0.315 | 0.945 ± 0.234 | 0.87 ± 0.176 |
| MiddlePhalanxTW | 0.556 ± 0.136 | 0.736 ± 0.239 | 0.556 ± 0.136 | 1.092 ± 0.269 | 0.977 ± 0.23 | 0.916 ± 0.241 |
| MiddlePhalanxOutlineAgeGroup | 0.556 ± 0.136 | 0.736 ± 0.239 | 0.556 ± 0.136 | 1.093 ± 0.27 | 0.976 ± 0.23 | 0.916 ± 0.241 |
| DistalPhalanxOutlineCorrect | 0.556 ± 0.235 | 0.769 ± 0.366 | 0.557 ± 0.236 | 0.954 ± 0.323 | 0.98 ± 0.397 | 0.899 ± 0.36 |
| Crop | 0.593 ± 0.317 | 0.796 ± 0.447 | 0.619 ± 0.337 | 0.856 ± 0.402 | 0.865 ± 0.396 | 0.998 ± 0.52 |
| DistalPhalanxTW | 0.603 ± 0.266 | 0.862 ± 0.438 | 0.604 ± 0.267 | 1.086 ± 0.393 | 1.075 ± 0.458 | 0.981 ± 0.403 |
| DistalPhalanxOutlineAgeGroup | 0.603 ± 0.266 | 0.862 ± 0.438 | 0.604 ± 0.267 | 1.086 ± 0.393 | 1.075 ± 0.458 | 0.981 ± 0.403 |
| PowerCons | 0.653 ± 0.267 | 0.85 ± 0.382 | 0.665 ± 0.28 | 0.946 ± 0.377 | 1.283 ± 0.5 | 1.244 ± 0.465 |
| ChlorineConcentration | 0.673 ± 0.118 | 1.064 ± 0.213 | 0.776 ± 0.141 | 0.677 ± 0.114 | 0.925 ± 0.164 | 0.933 ± 0.189 |
| FaceFour | 0.75 ± 0.125 | 1.072 ± 0.206 | 0.751 ± 0.126 | 0.892 ± 0.137 | 1.031 ± 0.171 | 1.016 ± 0.164 |
| MelbournePedestrian | 0.784 ± 0.795 | 1.031 ± 1.095 | 0.799 ± 0.806 | 2.096 ± 2.353 | 1.118 ± 1.117 | 1.483 ± 1.507 |
| FaceAll | 0.786 ± 0.309 | 1.025 ± 0.469 | 0.819 ± 0.347 | 0.895 ± 0.296 | 1.433 ± 0.53 | 1.38 ± 0.54 |
| FacesUCR | 0.79 ± 0.318 | 1.025 ± 0.469 | 0.82 ± 0.343 | 0.886 ± 0.319 | 1.433 ± 0.529 | 1.407 ± 0.531 |
| RefrigerationDevices | 0.892 ± 0.836 | 1.401 ± 1.449 | 0.97 ± 1.152 | 1.386 ± 0.969 | 1.44 ± 1.087 | 1.319 ± 1.006 |
| ECG200 | 0.959 ± 0.416 | 1.263 ± 0.647 | 1.018 ± 0.494 | 1.066 ± 0.388 | 1.262 ± 0.48 | 1.525 ± 0.597 |
| ElectricDevices | 1.251 ± 1.152 | 2.027 ± 1.89 | 1.463 ± 1.39 | 1.552 ± 1.144 | 1.721 ± 1.458 | 1.751 ± 1.483 |
| SonyAIBORobotSurface1 | 1.577 ± 0.451 | 2.182 ± 0.765 | 1.644 ± 0.498 | 1.863 ± 0.518 | 2.62 ± 0.694 | 2.719 ± 0.888 |
| ItalyPowerDemand | 1.671 ± 0.681 | 2.233 ± 1.078 | 1.71 ± 0.703 | 3.861 ± 1.039 | 2.546 ± 0.933 | 3.341 ± 1.111 |
| SonyAIBORobotSurface2 | 1.979 ± 0.462 | 2.817 ± 0.99 | 2.16 ± 0.567 | 2.155 ± 0.554 | 3.384 ± 0.752 | 3.89 ± 0.693 |
| Chinatown | 1.997 ± 0.438 | 2.764 ± 0.98 | 2.044 ± 0.462 | 3.849 ± 1.0 | 3.03 ± 0.617 | 3.855 ± 0.649 |
| DodgerLoopDay | 2.473 ± 0.363 | 3.681 ± 0.477 | 2.816 ± 0.414 | 2.475 ± 0.363 | 3.101 ± 0.433 | 3.425 ± 0.449 |
| DodgerLoopGame | 2.473 ± 0.363 | 3.681 ± 0.477 | 2.816 ± 0.414 | 2.475 ± 0.363 | 3.101 ± 0.433 | 3.425 ± 0.449 |
| DodgerLoopWeekend | 2.473 ± 0.363 | 3.681 ± 0.477 | 2.816 ± 0.414 | 2.475 ± 0.363 | 3.101 ± 0.433 | 3.425 ± 0.449 |
| CBF | 2.802 ± 0.467 | 4.215 ± 0.729 | 3.238 ± 0.543 | 2.819 ± 0.474 | 3.541 ± 0.526 | 3.972 ± 0.621 |
| TwoPatterns | 3.612 ± 0.662 | 5.228 ± 1.064 | 3.715 ± 0.679 | 4.843 ± 1.289 | 5.5 ± 0.828 | 5.419 ± 0.774 |
| ACSF1 | 4.452 ± 0.5 | 8.8 ± 1.022 | 5.758 ± 0.704 | 4.804 ± 0.427 | 4.65 ± 0.704 | 5.609 ± 0.582 |
| Earthquakes | 4.759 ± 1.075 | 6.754 ± 1.28 | 5.014 ± 1.084 | 5.886 ± 0.77 | 4.892 ± 1.175 | 5.333 ± 1.025 |
| SyntheticControl | 5.014 ± 1.957 | 7.801 ± 3.306 | 5.77 ± 2.374 | 5.087 ± 1.981 | 6.446 ± 2.42 | 7.577 ± 2.936 |
| SmoothSubspace | 9.628 ± 2.871 | 12.584 ± 4.351 | 10.064 ± 3.152 | 17.437 ± 4.913 | 11.338 ± 3.272 | 15.214 ± 3.689 |
| ShapeletSim | 12.409 ± 0.808 | 19.569 ± 1.165 | 14.394 ± 0.767 | 12.409 ± 0.808 | 15.407 ± 0.78 | 17.486 ± 0.962 |

## REFERENCES

[1] F. Eichinger, P. Efros, S. Karnouskos, and K. Böhm, "A time-series compression technique and its application to the smart grid," *VLDB J.*, vol. 24, no. 2, pp. 193–218, Apr. 2015.

[2] J. Azar, A. Makhoul, M. Barhamgi, and R. Couturier, "An energy efficient IoT data compression approach for edge machine learning," *Future Gener. Comput. Syst.*, vol. 96, pp. 168–175, Jul. 2019.

[3] S. Aljanabi and A. Chalechale, "Improving IoT services using a hybrid fog-cloud offloading," *IEEE Access*, vol. 9, pp. 13775–13788, 2021, doi: 10.1109/ACCESS.2021.3052458.

[4] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "An online algorithm for segmenting time series," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2001, pp. 289–296.

[5] D. Blalock, S. Madden, and J. Guttag, "Sprintz: Time series compression for the Internet of Things," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 2, no. 3, pp. 1–23, Sep. 2018.

[6] R. N. Bracewell and R. N. Bracewell, *The Fourier Transform and Its Applications*, vol. 31999. New York, NY, USA: McGraw-Hill, 1986.

[7] A. Graps, "An introduction to wavelets," *IEEE Comput. Sci. Eng.*, vol. 2, no. 2, pp. 50–61, Jun. 1995.

[8] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *Proc. 8th ACM SIGMOD Workshop Res. Issues Data Mining Knowl. Discovery (DMKD)*, 2003, pp. 2–11.

[9] J. W. Lin, S. Liao, and F. W. Leu, "Sensor data compression using bounded error piecewise linear approximation with resolution reduction," *Energies*, vol. 12, no. 13, p. 2523, Jun. 2019.

[10] M. Sifuzzaman, M. R. Islam, and M. Ali, "Application of wavelet transform and its advantages compared to Fourier transform," *J. Phys. Sci.*, vol. 13, no. 1, pp. 121–137, 2009.

[11] S. A. A. Karim, M. H. Kamarudin, B. A. Karim, M. K. Hasan, and J. Sulaiman, "Wavelet transform and fast Fourier transform for signal compression: A comparative study," in *Proc. Int. Conf. Electron. Devices, Syst. Appl. (ICEDSA)*, Apr. 2011, pp. 280–285, doi: 10.1109/ICEDSA.2011.5959031.

[12] J. Ledolter, "Smoothing time series with local polynomial regression on time," *Commun. Statist.-Theory Methods*, vol. 37, no. 6, pp. 959–971, Feb. 2008.

[13] C. Yang, X. Zhang, C. Zhong, C. Liu, J. Pei, K. Ramamohanarao, and J. Chen, "A spatiotemporal compression based approach for efficient big data processing on cloud," *J. Comput. Syst. Sci.*, vol. 80, no. 8, pp. 1563–1583, Dec. 2014.

[14] T. Pelkonen, S. Franklin, J. Teller, P. Cavallaro, Q. Huang, J. Meza, and K. Veeraraghavan, "Gorilla: A fast, scalable, in-memory time series database," *Proc. VLDB Endowment*, vol. 8, no. 12, pp. 1816–1827, Aug. 2015.

[15] R. Vestergaard, D. E. Lucani, and Q. Zhang, "A randomly accessible lossless compression scheme for time-series data," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Jul. 2020, pp. 2145–2154, doi: 10.1109/INFOCOM41043.2020.9155450.

[16] A. Jain and E. Y. Chang, "Adaptive sampling for sensor networks," in *Proc. 1st Int. Workshop Data Manage. Sensor Netw., Conjunct VLDB*, 2004, pp. 10–16.

[17] H. A. Dau, A. Bagnall, K. Kamgar, C.-C.-M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, and E. Keogh, "The UCR time series archive," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 6, pp. 1293–1305, Nov. 2019.

[18] P. Carbone, A. Katsifodimos, S. Ewen, V. Markl, S. Haridi, and K. Tzoumas, "Apache flink: Stream and batch processing in a single engine," *Bull. IEEE Comput. Soc. Tech. Committee Data Eng.*, vol. 36, no. 4, pp. 1–12, 2015.

[19] M. J. Watson, A. Liakopoulos, D. Brzakovic, and C. Georgakis, "A practical assessment of process data compression techniques," *Ind. Eng. Chem. Res.*, vol. 37, no. 1, pp. 267–274, Jan. 1998.

[20] S. Steinarsson, "Downsampling time series for visual representation," Ph.D. dissertation, Univ. Iceland, Reykjavík, Iceland, 2013.

[21] U. Jugel, Z. Jerzak, G. Hackenbroich, and V. Markl, "M4: A visualization-oriented time series data aggregation," *Proc. VLDB Endowment*, vol. 7, no. 10, pp. 797–808, Jun. 2014.

[22] P. Bae, K.-W. Lim, W.-S. Jung, and Y.-B. Ko, "Practical implementation of m4 for Web visualization service," *J. Commun. Netw.*, vol. 19, no. 4, pp. 384–391, Aug. 2017, doi: 10.1109/JCN.2017.000062.

[23] E. Fink and H. S. Gandhi, "Compression of time series by extracting major extrema," *J. Exp. Theor. Artif. Intell.*, vol. 23, no. 2, pp. 255–270, Jun. 2011, doi: 10.1080/0952813X.2010.505800.

[24] J. Liu, F. Chen, and D. Wang, "Data compression based on stacked RBM-AE model for wireless sensor networks," *Sensors*, vol. 18, no. 12, p. 4273, Dec. 2018.

[25] K. Kaindl and B. Steipe, "Metric properties of the root-mean-square deviation of vector sets," *Acta Crystallograph. A, Found. Crystallogr.*, vol. 53, no. 6, p. 809, 1997.

[26] T. Chai and R. R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)?" *Geosci. Model Develop. Discuss.*, vol. 7, no. 1, pp. 1525–1534, Feb. 2014.

[27] K. Jaskolka and A. Kaup, "Joint optimization of rate, distortion, and maximum absolute error for compression of medical volumes using HEVC intra," in *Proc. Picture Coding Symp. (PCS)*, Jun. 2018, pp. 126–130.

[28] T. Kimura, T. Kimura, A. Matsumoto, and K. Yamagishi, "Balancing quality of experience and traffic volume in adaptive bitrate streaming," *IEEE Access*, vol. 9, pp. 15530–15547, 2021, doi: 10.1109/ACCESS.2021.3052552.

**AMAIA GIL** received the degree in mathematics from the Faculty of Science and Technology, University of the Basque Country, Leioa, Spain, in 2014, and the master's degree in industrial mathematics from the Technical University of Madrid, Madrid, in 2016. She is currently pursuing the Ph.D. degree in computer science with the focus on the optimization of preprocessing steps for machine learning. She developed her End of Master Degree Project with Cidetec, about physical modeling of lithium pouch cells, in 2016. Since May 2016, she has been working with the Vicomtech Research Center, where she developed projects of the Smart Environment and Energy oriented to machine learning, visualization, and data science.

**MARCO QUARTULLI** received the Laurea degree in physics from the University of Bari, Italy, in 1997, and the Ph.D. degree in electrical engineering and computer sceince from the University of Siegen, Germany, in 2005. From 1997 to 2010, he worked on remote sensing ground segment engineering, image analysis, archives, and mining with Advanced Computer Systems, Italy. From 2000 to 2003, he was with the German Aerospace Center (DLR), Image Analysis Group, Remote Sensing Technology Institute, Germany, where he worked on metric resolution synthetic aperture radar image understanding in urban environments and content-based image retrieval. Since 2010, he has been with the Vicomtech, where he works in the Data Intelligence for Energy, Department of Industry and Environment.

**IGOR G. OLAIZOLA** received the degree in electronics engineering from the University of Navarra, Spain, in 2001, and the Ph.D. degree from the University of the Basque Country (UPV/EHU), Spain, in 2013. He is currently the Head of the Department of Data Intelligence for Industry-Energy, and Environment, Vicomtech, Spain. He has participated in more than 30 Research and Development Projects in the area of media technologies, signal/data analysis, and machine learning. Since 2013, he has been an Invited Lecturer with the University of Navarra. His current research interests include signal processing and methodologies to apply data exploitation techniques in industrial processes.

**BASILIO SIERRA** is currently a Full Professor with the Computer Sciences and Artificial Intelligence Department, University of the Basque Country. He is also the Co-Director of the Robotics and Autonomous Systems Group, Donostia-San Sebastian. He is also a Researcher in robotics and machine learning, where he is working on the use of different paradigms to improve behaviours. His research interests include machine learning, data analysis, computer vision, and robotics.