sensors

Article

Efficient Kernel-Based Subsequence Search for Enabling Health Monitoring Services in IoT-Based Home Setting

Antonio Candelieri *, Stanislav Fedorov and Enza Messina

Department of Computer Science, Systems and Communication, University of Milano-Bicocca, 20126 Milano MI, Italy; fedorov.stas.wrk@gmail.com (S.F.); enza.messina@unimib.it (E.M.)

* Correspondence: antonio.candelieri@unimib.it; Tel.: +39-0264487926

Received: 30 September 2019; Accepted: 23 November 2019; Published: 27 November 2019

Abstract: This paper presents an efficient approach for subsequence search in data streams. The problem consists of identifying coherent repetitions of a given reference time-series, also in the multivariate case, within a longer data stream.The most widely adopted metric to address this problem is Dynamic Time Warping (DTW), but its computational complexity is a well-known issue. In this paper, we present an approach aimed at learning a kernel approximating DTW for efficiently analyzing streaming data collected from wearable sensors, while reducing the burden of DTW computation. Contrary to kernel, DTW allows for comparing two time-series with different length. To enable the use of kernel for comparing two time-series with different length, a feature embedding is required in order to obtain a fixed length vector representation. Each vector component is the DTW between the given time-series and a set of "basis" series, randomly chosen. The approach has been validated on two benchmark datasets and on a real-life application for supporting self-rehabilitation in elderly subjects has been addressed. A comparison with traditional DTW implementations and other state-of-the-art algorithms is provided: results show a slight decrease in accuracy, which is counterbalanced by a significant reduction in computational costs.

Keywords: data stream analysis; pattern query; kernel learning; dynamic time warping; subsequence search

1. Introduction

Dynamic Time Warping [1] is a technique to find the optimal alignment between two time-series, by considering the possibility to nonlinearly "warp" one time-series by stretching or shrinking it along its time axis. The amount of warping needed for the alignment is then used as a measure of the difference between the two time-series. A typical application of DTW is speech recognition [2], where it is used to determine if two waveforms represent the same spoken phrase. In a speech waveform, the duration of each spoken sound and the interval between sounds can vary, but the overall speech waveforms must have a similar "shape". In addition to speech recognition, DTW has also been found useful in many other disciplines, including, gesture recognition [3], robotics [4], manufacturing [5], and health monitoring [6–8].

Moreover, measuring the similarity between two time-series is a core task for time-series clustering [9], where both data representation and preprocessing are critical choices, as well as the definition of a suitable similarity measure. Recently, in [10], a fuzzy-clustering approach for time-series data has been proposed, where DTW distance is used for comparing pairs of time-series. Other relevant approaches, which could benefit of a more efficient DTW computation, are [11,12].

Indeed, despite its widely adoption in many application domains, a well-known issue of DTW is its computational complexity. Computing DTW between two time-series requires $O(NM)$, where N and M are the lengths of the two time-series. Therefore, the comparison of a reference time-series with a large data stream (i.e., $M \gg N$), as well as the identification of all the reference repetitions within the data stream, might be computationally very expensive.

The importance of the topic is widely recognized in the scientific community, as highlighted by relevant previous works quoted in Section 2.

The specific contributions of this paper are as follows.

- Designing a kernel learning task aimed at approximating DTW to reduce computational burden of the subsequence search.
- Comparing the proposed kernel-based DTW approximation with traditional DTW-based implementations and other state-of-the-art algorithms.
- Validating the proposed approach on a simple benchmark toy example (https://www.cs.unm.edu/mueen/FastestSimilaritySearch.html) and on a more complex one, namely, the "User Identification From Walking Activity" dataset, freely downloadable from the UCI Repository.
- Validating the results through pattern query experiments on a dataset self-rehabilitation dataset, specifically collected in a real-life project. Self-rehabilitation dataset is available from the corresponding author on reasonable request.

The rest of the paper is organized as follows. Section 2 provides the methodological background about DTW, its recent innovations and applications, as well as computational drawbacks in the specific case of subsequence search. Section 3 describes how to learn a kernel to approximate DTW and, consequently, increase the computational efficiency subsequence search within long data streams. Section 4 presents the experimental setting and the datasets used to validate the approach. Section 5 summarizes the experimental results. Finally, the discussion and relevant conclusions are reported in Section 6.

2. Backgound

2.1. Dynamic Time Warping

The core component of DTW is a data structure named "accumulated cost matrix", denoted by $D \in \mathbb{R}^{N \times M}$, where N and M are the lengths of the two time-series, $X = (x_1, \dots, x_N)$, and $Y = (y_1, \dots, y_M)$ to be compared. Every entry $D_{i,j}$ of this matrix is computed as follows,

$$D_{i,j} = \min\{D_{i-1,j-1}, D_{i-1,j}, D_{i,j-1}\} + c(x_i, y_j) \;\; \forall i = 1, \dots, N \;\; \mathit{and} \;\; j = 1, \dots, M, \tag{1}$$

where x_i and y_j are the i-th and j-th values of X and Y, respectively, and $c : X \times Y \rightarrow \mathbb{R}_0^+$ is a cost function. The most widely adopted cost function is the Euclidean distance. The initialization can be simplified by extending the accumulated cost matrix D with an additional row and column, specifically $D_{i,0} = \infty$, $D_{0,j} = \infty$, and $D_{0,0} = 0$. Then, the recursion in Equation (1) holds for $i = 1, \dots, N$ and $j = 1, \dots, M$.

In the general case, the two time-series could be multivariate, and consequently x_i and y_j could be vectors. Furthermore, the accumulated cost matrix satisfies, by construction, the following identities,

$$D_{i,1} = \sum_{k=1}^{i} c(x_k, y_1) \;\; \forall i = 1, \dots, N \tag{2}$$

and

$$D_{1,j} = \sum_{k=1}^{j} c(x_1, y_k) \;\; \forall j = 1, \dots, M \tag{3}$$

A warping path is defined as a sequence $p = p_1, \ldots, p_L$ of positions in D, where $p_l = (i_l, j_l)$. The warping path represents an alignment between X and Y, which assigns the element x_{i_l} of the first series to the element y_{j_l} of the second series. A warping path must satisfy the following conditions.

- Boundary conditions $p_1 = (1,1)$ and $p_L = (N, M)$. These conditions enforce the alignment to start and finish at the extremes of the two series, meaning that the first elements of X and Y, as well as the last ones, must be aligned to each other.
- Monotonicity condition: $i_1 \leq i_2 \leq \ldots \leq i_L$ and $j_1 \leq j_2 \leq \ldots \leq j_L$. This condition simply ensures that if an element in X precedes a second one this should also hold for the corresponding elements in Y, and vice versa.
- Step size condition: $p_{l+1} - p_l \in (1,0), (0,1), (1,1)$ for $l = 1, \ldots, L-1$. This condition ensures that no element in X and Y can be omitted and that there are no replications in the alignment, meaning that all the index pairs contained in a warping path are pairwise distinct. Note that the step size condition implies the monotonicity condition.

The total cost, $c_p(X,Y)$, associated to a warping path p between X and Y is computed as

$$c_p(X,Y) = \sum_{l=1}^{L} c(x_{i_l}, x_{i_l})$$

An optimal warping path between X and Y is a warping path, p^*, having minimal total cost over all the possible warping paths, as the one one shown in Figure 1. Therefore, the DTW distance between X and Y is obtained as

$$DTW(X,Y) = c_{p^*}(X,Y) = \min_{p} c_p(X,Y) \tag{4}$$

Note that DTW is symmetric if the cost function $c(\cdot,\cdot)$ is symmetric. However, DTW is generally not positive definite and does not always satisfy the triangle inequality. The following figure provides an example of accumulated cost matrix and the associated optimal warping path. The closer the optimal warping path to the diagonal, the lower the misalignment between the two series.

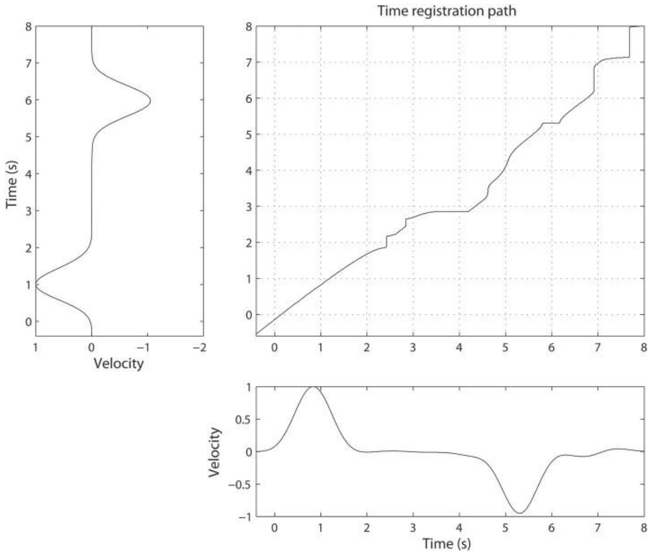

Figure 1. An illustration of accumulated cost matrix and associated optimal warping path when using DTW to align the two time-series in the picture.

The optimization problem (4) is solved using dynamic programming, with complexity $O(NM)$. The optimal warping path algorithm is summarized in the following Algorithm 1.

Algorithm 1 Optimal warping path algorithm

```
Input  Accumulated cost matrix $D$
Output  optimal warping path $p^*$
$i \leftarrow N, j \leftarrow M, p_1 \leftarrow (i,j), l \leftarrow 1$
while $i \neq 1$ OR $j \neq 1$ do
    $l \leftarrow l + 1$
    if $i = 1$ then
        $p_l \leftarrow (i, j-1)$
        $j \leftarrow j - 1$
    else
        if $j = 1$ then
            $p_l \leftarrow (i-1, j)$
            $i \leftarrow i - 1$
        else
            $d^* \leftarrow \min\{D_{i-1,j-1}, D_{i-1,j}, D_{i,j-1}\}$
            if $d^* = D_{i-1,j}$ OR $d^* = D_{i-1,j-1}$ then
                $i \leftarrow i - 1$
            end if
            if $d^* = D_{i,j-1}$ OR $d^* = D_{i-1,j-1}$ then
                $j \leftarrow j - 1$
            end if
            $p_l \leftarrow (i,j)$
        end if
    end if
end while
$p^* = reverse(p)$
```

The reverse operation at the end of the algorithm is necessary because the length L of the optimal warping path p^* is unknown a priori. Indeed, the optimal warping path is computed, according to the dynamic programming paradigm, in a reverse order starting from the position (N, M) to the position $(1, 1)$. Therefore, the reverse operation allows to give, as output, the optimal warping path, represented as a sequence of positions coherent with the initial definition.

A commonly adopted DTW variant is to impose global constraint conditions on the admissible warping paths with the aim to prevent undesired alignments by controlling the route of a warping path. Two widely adopted global constraints are the Sakoe–Chiba band [13] and the Itakura parallelogram [14]. Besides the prevention of undesired alignments, global constraints can also speed up DTW computation, because they in effect limit the length, L, of p^*.

A review of the research efforts in optimizing both the efficiency and effectiveness of DTW-based algorithms for similarity search, clustering, and classification is presented in [15]. Here, different variants of DTW are discussed, such as constrained DTW, multidimensional DTW, asynchronous DTW, along with optimization techniques for improving DTW efficiency, such as lower bounding, early abandoning, run-length encoding, bounded approximation, and hardware optimization. Some relevant aspects of DTW optimization are presented in [16], where an example of approximation of the accumulated cost matrix D is given, and in [17], where the distance calculations for univariate DTW are accelerated. In [18], a exotic approach to increase robustness of similarity measure by constructing a matrix over the derivative approximation of neighborhood samples can be found, namely, Derivative DTW. Finally, with respect to the topic of DTW approximation via kernel, the authors of [19] propose a kernel aimed at learning the principal global alignments for the given data by using the hidden structure of the alignments from the training data. This approach is presented as more computationally efficient when compared to previous kernels on DTW distance, such as GA kernel [20,21] and Gaussian DTW kernel [22].

2.2. Dynamic Time Warping for Subsequence Search

The problem of subsequence search, based on DTW, is described in [1], and is also known as subsequence DTW or DTW-based subsequence search. In this task, the time-series to be compared are characterized by significant difference in their lengths, i.e., $M \gg N$, where the shortest one is called reference pattern, that is, a specific sequence to be searched for within the second longer time-series. Indeed, instead of searching for a global alignment between the two series, the goal is to find at least one subsequence of the reference pattern within the longer data stream, with optimal fitting (i.e., minimal DTW). Let us denote by a^* and b^* the indices representing the beginning and end of the subsequence within Y, with $1 \leq a^* \leq b^* \leq M$. These indices are identified by solving the following optimization problem,

$$(a^*, b^*) = \underset{(a,b):1 \leq a^* \leq b^* \leq M}{\arg\min} DTW(X, Y_{a:b}), \tag{5}$$

where $Y_{a:b}$ is the subsequence $(y_a, \ldots, y_b)$ in Y.

Optimization problem (5) can be solved by applying a modification in the initialization of the previous DTW algorithm, consisting of replacing (3) with

$$D_{1,j} = c(x_1, y_j)$$

In other words, contrary to the identities in Equation (3), the starting position of the subsequence a^* does not provide any value, except its own cost, and therefore the cost of positioning b^* depends only on the DTW between the reference pattern and the chosen subsequence. The remaining values of the accumulated cost matrix D are defined as in the basic DTW algorithm.

The index b^* is determined as $b^* = \arg\min_{b=1,\ldots,M} D_{N,b}$. In case b^* is not unique, the lexicographic order can be used to select among the multiple choices. Given the value b^*, then a^* is obtained by applying the optimal warping path algorithm, starting from position (N, b^*). Finally, the resulting optimal warping path $p^* = (p_1, \ldots, p_L)$ must be reduced to $(p_l, \ldots, p_L)$, where p_l is the maximum index such that $p_l = (a^*, 1)$, with $l \in \{1, \ldots, L\}$. Therefore, the optimal warping path between X and $Y_{a^*:b^*}$ is given by $(p_l, \ldots, p_L)$, and, roughly speaking, all the elements preceding y_{a^*} and those following y_{b^*} are not considered in the alignment and, consequently, do not account for additional costs to DTW.

In the following, we summarize how the subsequence search algorithm can be extended to find multiple repetitions of the reference pattern X within the longer data stream Y. First, we introduce, as reported in [1], the distance function $\Delta : [1 : M] \rightarrow R$, with $\Delta(b) = D_{N,b}$ $b = 1, \ldots, M$, which assigns the minimal DTW that can be computed between the reference pattern X and a subsequence in Y ending in y_b. Given b, the starting index y_a of the searched subsequence is identified through the optimal warping path algorithm revised for subsequence search. The procedure is summarized in Algorithm 2.

Step 8 of the Algorithm 2 is particularly important. As the elements of the accumulated cost matrix are computed accordingly to Equation (1), we set $\Delta(b) = \infty$ in the neighborhood of the optimal value b^* to avoid subsequences already found (optimization process in step 4) as well as pathological cases of a very short time-series in its neighborhood.

To guarantee no false dismissals in similarity query processing and efficiently prune a significant number of the search candidates, leading to a reduction in the search cost, the authors of [23] propose a fast similarity search method (FTW).

Algorithm 2 DTW-based subsequence search algorithm

```
Input  reference pattern $X = (x_1, \dots, x_N)$, a longer data stream $Y = (y_1, \dots, y_M)$, with $M \gg N$, and a threshold $\tau$
Output  a list $\mathcal{L}$ of repetitions of $X$ within $Y$ having, individually, a DTW lower than $\tau$. The list is ranked depending on the individual DTW
1: $\mathcal{L} = \emptyset$
2: compute the accumulated cost matrix $D$ between $X$ and $Y$
3: compute the distance function $\Delta$
4: find $b^* = \arg\min_{b \in \{1,\dots,M\}} \Delta(b)$
5: If $\Delta(b) > \tau$ then  STOP
6: find $a^*$ by using Optimal warping path algorithm but initialized in $j = b$ instead of $j = M$
7: updating $\mathcal{L}$ as: $\mathcal{L} = \mathcal{L} \cup Y_{a^*:b^*}$
8: Set $\Delta(b) = \infty$ for every $b$ in a suitable neighborhood of $b^*$
9: GO TO STEP 3
```

3. Learning a Kernel to Approximate DTW

3.1. Time-Series Kernels via Alignments

A number of global alignment kernels have been proposed in literature with the aim to extend DTW to a kernel-based estimation method. The underlying idea is to avoid the problem of searching exactly the optimal warping path while learning a kernel approximating the DTW value between two time-series. Kernel methods have shown to be promising for learning complex models by implicitly transforming a simple representation, like mapping typical Euclidean distance into a high-dimension feature space [24]. The main obstacles for applying usual kernel methods to time-series are due to two distinctive characteristics of time-series: (a) variable length and (b) dynamic time scaling and shifts. Furthermore, direct use of DTW leads to a not positive definite kernel that does not provide a convex optimization problem [20]. To overcome these obstacles, a family of global alignment kernels have been proposed by taking soft-max over all possible alignments in DTW to give a positive definite kernel [20,21,25]. However, the effectiveness of the global alignment kernels is impaired by the diagonal dominance of the resulting kernel matrix proportional to the difference in the size between the two time series [21], which is the case of subsequence search. In [26], a random features mapping method for time-series embedding is proposed: the idea is to use an explicit mapping to represent any time-series through its alignments to a set of randomly chosen "basis times-series", having a small length. This significantly reduces computational cost. Starting from similar considerations, our approach aims at learning a kernel, based on random features mapping, to use for efficiently solving the subsequence search problem.

3.2. Learning a Kernel for Subsequence Search

Consider two multimodal time-series $X = (x_1, \dots, x_N)$ and $Y = (y_1, \dots, y_M)$ with $M \gg N$, where X represents the reference pattern to be searched in Y. Let us now randomly generate a sequence $S = \{s_1, s_2, \dots, s_R\}$ of R basis time-series, where $s_i \in \mathbf{R}^{d \times L_i}$ $\forall i = 1, \dots, R$, d-dimension of data and $L_i \in [L_{\text{min}}, L_{\text{max}}]$ is the length of the i-th basis time-series s_i (usually $L_i \ll M$ and $L_i < N$), and where L_{min} and L_{max} are the minimum and maximum length allowed. R, L_{min}, and L_{max} are technical parameters of the algorithm and must be tuned experimentally. The considerations on computational complexity given in the following provide useful relations for setting up the values of these parameters suitably.

According to the authors of [26], if the set S of basis time-series is sampled from a normal distribution, it shows good performance in further construction of kernel.

Let us define the feature map $\mathbf{\Phi}_S(X) = (\phi_{s_1}(X), \ldots, \phi_{s_R}(X))^T$, where the i-th component of $\mathbf{\Phi}_S(X)$ is the alignment between X and the random series s_i. We consider DTW as measure of this alignment, thus we must compute the accumulated cost matrix D for every entry of the feature vector:

$$\mathbf{\Phi}_S(X) = (DTW(X, s_1), \ldots, DTW(X, s_R))^T \tag{6}$$

The mapping (6) provides an R dimensional vector without correspondence to dimensionality of original time series, therefore it is used in the further construction of a kernel able to work with time-series having, originally, different length. Although DTW must be computed R times, by keeping $RL_{\max} < N$ the computational cost is low due to the reduced length of each s_i, more specifically. Indeed, the cost for computing $\mathbf{\Phi}_{s_i}(X)$, in the worst case, is $O(NRL_{\max})$. Moreover, parallel computing can be used to further improve efficiency, since the computation of each component of the vector $\phi_S(X)$ is embarrassingly parallel, as depicted in the following figure, where we assume, for sake of simplicity, to have R different processors available.

Given the two time-series X and Y, $\mathbf{\Phi}_S(X)$ and $\mathbf{\Phi}_S(Y)$ are their associated representation in the space spanned by their DTW with respect to the set of basis series S. Note that whichever is the length of X and Y, their mappings, $\mathbf{\Phi}_S(X)$ and $\mathbf{\Phi}_S(Y)$, have the same length, that is, R. Contrary to the authors of [26], we decide to preserve the same set of basis series, S, that lead us to equality of spanned spaces and let us fairly compare time-series regardless of the diagonal dominance problem given by skewed data. To introduce the positive definite distance, we decided to use a nonlinear kernel, i.e., Radial Basis Function (RBF) kernel, comparing $\mathbf{\Phi}_S(X)$ and $\mathbf{\Phi}_S(Y)$:

$$K(\mathbf{\Phi}_S(X), \mathbf{\Phi}_S(Y)) = \exp(-\frac{1}{2\gamma^2}\|\mathbf{\Phi}_S(X) - \mathbf{\Phi}_S(Y)\|^2)$$

where $\|\cdot\|$ denotes Euclidean norm and γ is the length-scale parameter. Although a simple linear kernel could be adopted, such as in [26], this reduces the approximating capability of the approach. On the other hand, the adoption of the RBF kernel leads to the need for optimizing the length-scale hyperparameter.

As kernel measures the similarity between the two time-series, we used the following formula to define our DTW kernel-based distance,

$$d_K(X, Y) = 1 - K(\mathbf{\Phi}_S(X), \mathbf{\Phi}_S(Y))$$

The computational complexity of $d_K(X, Y)$ is given by the computational complexity of $\mathbf{\Phi}_S(X)$ and $\mathbf{\Phi}_S(Y)$, leading to $O(RL_{max}N + RL_{max}M)$. Let us denote $L_{max} = \alpha N$, with $\alpha \in (0, 1)$; then, to be more efficient than traditional DTW, the following relation must be satisfied,

$$O(RL_{max}N + RL_{max}M) = \rho O(NM), \tag{7}$$

with $\rho \in (0, 1)$. From Equation (7) it is possible to derive the relation linking the technical parameters of the proposed approach. More precisely,

$$RL_{max}N + RL_{max}M = \rho NM$$

and consequently

$$R = \frac{\rho M}{\alpha(N + M)} \tag{8}$$

where $\lfloor R \rfloor$ represents the maximum cardinality of S, given the value of ρ and α.

Now, two different cases are to be considered: if $N \simeq M$, i.e., using the kernel-based DTW approximation to compare two series with similar length, then $R = \rho/2\alpha$; whereas if $N \ll M$, i.e., searching for subsequence in a longer data stream, then $R \simeq \rho/\alpha$. It is interesting to note that, in both two cases, the value of R does not depend on the lengths N and M. Let us consider a simple example where $\alpha = 0.1$ and $\rho = 0.5$, meaning that we want a reduction of computational cost of 50% with respect to DTW and to use a basis series no longer than 10% of the reference time-series X ($L_{max} = \alpha N$). Then, according to Equation (8), we obtain $R = 5$. This value might be too small to obtain a good DTW approximation. However, thanks to the proposed parallel computation schema, it is possible to increase the number of basis series up to $\bar{R} = n_{pp} \lfloor R \rfloor$, with n_{pp} being the number of parallel processes. For instance, with $n_{pp} = 6$, parallel processors can use the $\bar{R} = 30$ basis time-series, which can be considered statistically significant.

With respect to the subsequence search problem, it is now possible to replace DTW in Equation (5) with its kernel-based approximation:

$$\begin{aligned} &(a^*, b^*) = \underset{(a,b):1 \le a < b \le M}{\arg\min} \; d_K(X, Y_{a:b}) \\ &\text{subject to:} \\ &b - a \le \beta^+ N \\ &b - a \ge \beta^- N \end{aligned} \tag{9}$$

where β^+ and β^- are two coefficients to set up, representing, respectively, the largest and the smallest length of the possible subsequence with respect to N, that is, the length of the reference time-series X. Values of these coefficients depend on the specific application: suitable ranges are $\beta^+ \in [1, 2)$ and $\beta^- \in (0, 1]$.

Due to the nature of $d_K(X, Y)$, which requires the computation of $\Phi(Y_{a:b})$ for each pair (a, b), Equation (9) is a Black-Box Optimization (BBO) problem. We solve it via Bayesian Optimization (BO) [27]. BO is a technique successfully applied for automating the configuration of Machine Learning algorithms, such as autoML [28,29], as well as complex Machine Learning pipelines [8]. The aim is to obtain a good solution of (9) by trying a limited number of possible pairs (a, b), i.e., function evaluations.

Solving Equation (9) via BO requires $O(\alpha R N^2 + \alpha R N \beta^+ N \eta) + O_{BO}$, where $\beta^+ N$ is the maximum length of the subsequence $Y_{a:b}$, η is the maximum number of function evaluations, and the term O_{BO} summarizes the computational cost of BO. This cost is usually dominated by $O(t^3)$ in the case of Gaussian processes-based BO, where t is the number of function evaluations performed ($t = 1, \ldots \eta$), so it increases with the number of pairs (a, b) evaluated. In any case, η can be chosen to be $\eta \ll N$.

To be more efficient than DTW-based subsequence search, the following relation must be satisfied,

$$O(\alpha R N^2 + \alpha R N \beta^+ N \eta) + O_{BO} < O(MN) + O_{owp} \tag{10}$$

where O_{owp} summarizes the complexity of the optimal warping path algorithm (Algorithm 1). For this analysis, we can consider negligible both O_{BO} and O_{owp}. Therefore,

$$\alpha R N^2 + \alpha R N \beta^+ N \eta < MN \tag{11}$$

One can now set

$$\alpha R N^2 + \alpha R N \beta^+ N \eta = \bar{\rho} M N \tag{12}$$

with $\bar{\rho} \in (0, 1)$, and consequently

$$\eta = \left\lfloor \frac{\bar{\rho} M - \alpha R N}{\alpha R N \beta^+} \right\rfloor \tag{13}$$

For example, consider two time-series X and Y having length $N = 100$ and $M = 10000$, respectively. If $\alpha = 0.1$ and $\rho = 0.5$, it follows from Equation (8) that $R = 5$. By further setting $\bar{\rho} = 0.5$ and $\beta^+ = 1.5$, it follows from (13) that $\eta = 66$, providing an upper bound on the number of function evaluations to perform during BO.

3.3. Extension to Multiple Reference Patterns

This section summarizes how we can extend our approach to implement a subsequence search when multiple reference patterns, X_i, with $i = 1, \dots, n$, have to be searched for within multiple data streams Y_j, with $j = 1, \dots, m$. The first step involves generating a specific kernel for each reference pattern by learning the best value of the length-scale γ_i directly from the data:

$$\gamma^* = \underset{\gamma \in \mathcal{R}^n}{\arg\min} \frac{2}{n(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} |(1 - K_{\gamma_i}(\phi_S(X_i), \phi_S(X_j)) - DTW(X_i, X_j))| \tag{14}$$

where $\gamma = (\gamma_1, \dots, \gamma_n)^T$ is the vector of the length-scale values of the kernels associated to the corresponding reference patterns $X_i, i = 1, \dots, n$. The idea is to minimize, for each possible pairs of time series X_i and X_j, the difference between the DTW and its kernel approximation computed on the mapping of the two time series, that is, $\phi_S(X_i)$ and $\phi_S(X_j)$. Given that the matrix is symmetric by construction, problem (14) can be reduced to minimize the upper triangular matrix of the errors between each pair of reference patterns.

As the range of RBF kernel is $[0, 1]$, DTW is preliminary rescaled in the same interval, as reported in step 8 of the following algorithm. Algorithm 3's parameters have to be tuned manually, according to the procedure previously described.

Algorithm 3 Learning a kernel for approximating DTW in the case of multiple references and multiple data streams

```
Input $n$ reference patterns $\{X_i\}_{i=1 \dots n}$, $m$ data streams $\{Y_l\}_{l=1 \dots m}$ and parameters $R$, $L_{\min}$, $L_{\max}$, $\beta^-$, $\beta^+$, $\sigma^2$.
Output a matrix $\mathbf{K} \in \mathbb{R}^{n \times m}$ containing the kernel values.
 1: generate a set of $R$ "basis" time series $S = \{s_1, s_2, \dots, s_R\}$, where each $s_i \sim \mathcal{N}(0, \sigma^2)$
 2: for $i = 1 : n$ do
 3:     compute $\mathbf{\Phi}_S(X_i) = (DTW(X_i, s_1), \dots, DTW(X_i, s_R))^T$
 4:     for $j = 1 : n$ do
 5:         $\mathbf{M}_{i,j} = DTW(X_i, X_j)$
 6:     end for
 7: end for
 8: normalize entries of the matrix $\mathbf{M}$
 9: choose $\gamma^* = \arg\min_{\gamma \in \mathbb{R}^n} \frac{2}{n(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} |(1 - K_{\gamma_i}(\phi_S(X_i), \phi_S(X_j)) - \mathbf{M}_{i,j})|$
10: compute every entry of $\mathbf{K}$ as $K_{i,l} = d_{K,\gamma^*}(X_i, Y_{l,a^*:b^*})$, where $a^l, b^*$ are obtained solving (9).
```

4. Experimental Setting

4.1. Organization of the Experiments

To validate the proposed kernel-based DTW approximation, we have considered three different experimental settings with different levels of complexity. More in detail, the first experiment considers the simplest case, that is, a univariate setting, where a given reference pattern is searched for within a longer data stream. The second experiment extends the analysis to a multivariate setting with multiple reference patterns. Finally, the third experiment refers to a real-life application (i.e., self-rahabilitation at home).

4.2. Experiment 1: A Univariate Case

The first experiment considers a univariate benchmark dataset consisting of accelerometer data collected on a Sony AIBO robot dog. The reference pattern to be searched (consisting of 100 data points) is related to acceleration collected when the dog was walking on a carpet, whereas the longer data stream refers to a sequence of data collected in three different conditions while robot was walking on cement (for 5000 data points), on carpet (for 3000 data points), and again on cement (for

5000 data points). Both the reference patterns and the data stream can be downloaded for free from https://www.cs.unm.edu/mueen/FastestSimilaritySearch.html. The goal is to search for the first 10 and 25 best-matching repetitions of the reference pattern into the data stream. Our algorithm was compared with three other approaches:

- MASS [30]: a fast similarity search algorithm for subsequences under Euclidean distance and correlation coefficient (experiments refer to MASS under Euclidean distance, only). A strong assumption of MASS is that the identified subsequences have the same length of the reference.
- DTW with fixed window: based on the same assumption of MASS but using DTW instead of Euclidean distance.
- DTW-based subsequence search algorithm described in Algorithm 2.
- DTW-based Kernel constructed to approximate the exact DTW.

4.3. Experiment 2: A Multivariate Case

The dataset we considered for this experiment is a benchmark dataset for user identification from walking activity [31], which can be freely downloaded from the UCI Repository website: (https://archive.ics.uci.edu/ml/datasets/User+Identification+From+Walking+Activity). The dataset refers to accelerometer data (i.e., acceleration on the x, y, and z axes) acquired through an Android smartphone positioned in the chest pocket and from 22 participants walking in the wild over a predefined path. Data information:

- Sampling frequency of the accelerometer: DELAY_FASTEST with network connections disabled.
- A separate file for each participant.
- Every row in each file consists of time-step, x acceleration, y acceleration, and z acceleration.

From the 22 data streams—one for each participant—a small portion of data (200 data points, that is, ~6 s) is extracted and considered as reference pattern for the corresponding user. The size of the reference pattern has been selected after some preliminary exploratory analysis on the entire set of recordings. Given a data stream, the goal is to associate it to the corresponding user. To do this, the data stream is associated to the user whose reference pattern results in the highest number of best matching repetitions. This the experiment is specifically devoted to test the Algorithm 3.

4.4. Experiment 3: A Real-Life Application

This dataset was specifically collected for designing and developing a digital service for supporting self-rehabilitation at home. We used three Inertial Measurement Units (IMUs) worn over the chest, the wrist (of the dominant arm), and the ankle (of the dominant leg), respectively. The sensors permit to acquire several measures over the three axes (i.e., orientation, acceleration, and velocity) with a frequency of 10 Hertz. For the purpose of this study, we considered the acceleration measures only, as they are the most relevant information about the movement performed by the subject. Data refers to an over 60-year-old woman performing the following schedule of five exercises.

- Flexo-extension of the knee (sit-down position)
- Raise and lower the arms (sit-down position)
- Rotate the torso (sit-down position)
- Back extension of the legs (stand-up position)
- Light squat (stand-up position)

These rehabilitation exercises were initially performed by the subject in the clinical setting under the supervision of a qualified trainer: the resulting reference patterns, certified by the trainer, are assumed as gold standards. Then, the subject performed a self-rehabilitation session at home and acceleration data were collected from the wearable sensors. Within the collected data stream, we searched for each one of the five reference patterns.

The correct identification of reference patterns was validated via visual inspection. Note that the exercises schedule (i.e., order and number of repetitions) was planned before the self-rehabilitation session, making it easy for the clinician to identify which part of the data stream corresponds to a specific exercise. The repetitions identified by the algorithms—traditional DTW and the kernel-based approximation—which are in the correct portion of data were considered as hit; otherwise, miss.

4.5. Computational Setting

All the experiments have been performed on an Intel Core i7-7700HQ CPU at 2.80 GHz, 16 GB RAM, Windows 10 OS (64 bit). Software used: R (3.6.0) and Matlab (R2019a).

As we are interested in comparing our approach with traditional DTW, which is not parallel, we do not exploit the parallelization schema reported in Section 3.2, Figure 2. Thus, time reduction reported in the results could be even more relevant if parallelization is used.

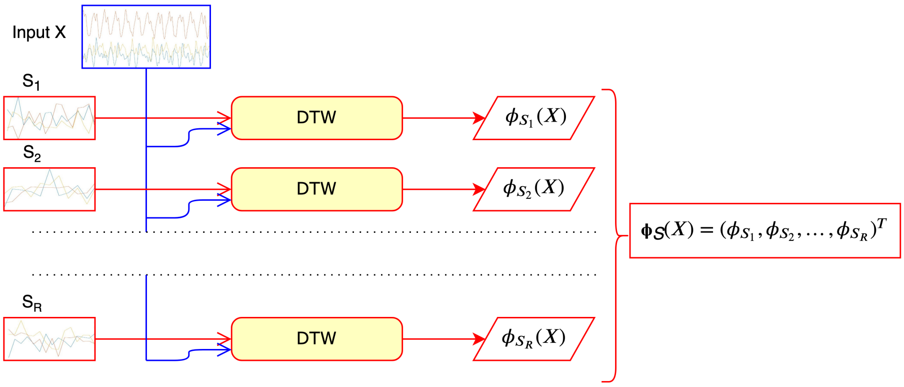

Figure 2. Illustration of the parallel computation of the components of the vector $\mathbf{\Phi}_S(X)$. Time-series in the figure are assumed multivariate.

5. Results

5.1. Results of Experiment 1

Although all the algorithms correctly identified the first 10 best-matching repetitions (Figure 3) within the portion of data collected while the robot dog was walking on the carpet, there remain some slight differences between the DTW-based Subsequence Search and MASS algorithm. Both the DTW-based subsequence search (i.e., Algorithm 2) and the proposed kernel-based approach allow temporal deformation or subsequences with a different length (i.e., the subsequence can be faster or slower, still maintaining the same shape of the reference pattern). Finally, the proposed kernel-based DTW approximation reduces the computational time more than 2 times (3.4 s compared to 8.1 s of Algorithm 2).

Of greater interest, when the number of repetitions to be found is increased from 10 to 25, the results provided by the four approaches are significantly different, as depicted in Figure 4. MASS and the DTW with fixed window size have identified, respectively, 9 and 6 subsequences outside the portion of data collected while the AIBO robot was walking on carpet. On the contrary, only four out of 25 subsequences identified by the DTW-based Subsequence Search (Algorithm 2) are outside the correct portion of data, and thus more effectively manage the temporal deformation between the subsequence and reference pattern by this algorithm, generate a more accurate identification. The proposed kernel-based DTW approximation results in a higher number of errors than DTW-based subsequence search (i.e., 8 subsequences outside the correct portion of data), that is, lower than MASS and comparable to DTW with fixed windows. Errors are due to the approximating nature of the approach but are counterbalanced by the significant reduction in computational time.

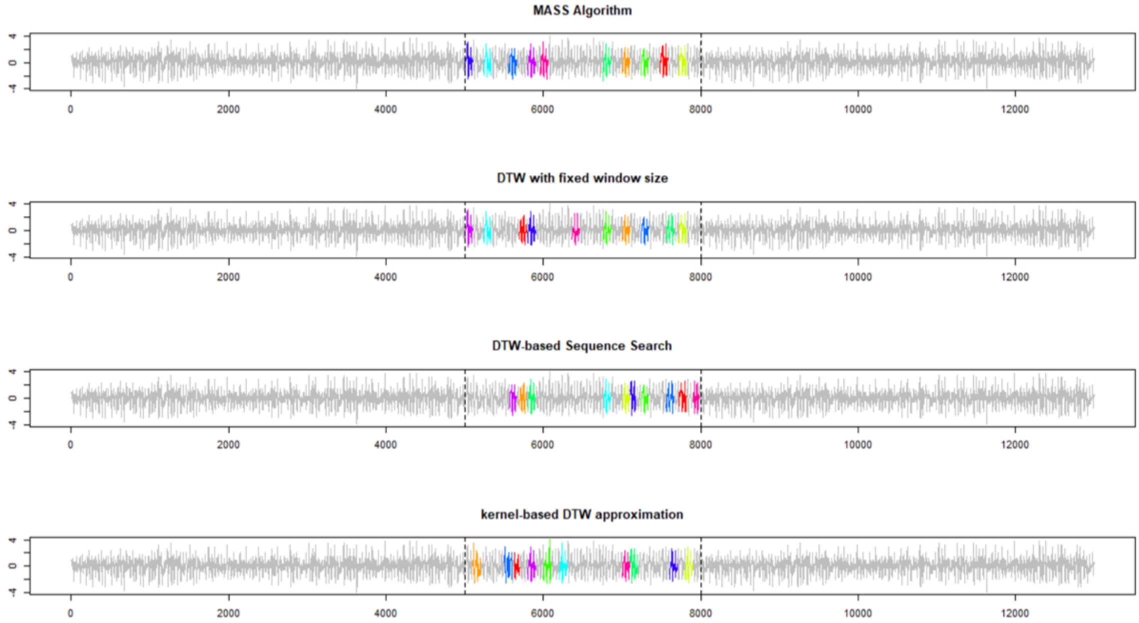

Figure 3. The first 10 best-matching subsequences were identified by four different algorithms on the Experiment 1. The reference pattern consists of 100 acceleration data points collected when a Sony AIBO robot was walking on a carpet. The data stream consists of 5000 data points when the same dog was walking on cement, then followed by 3000 walking on a carpet and then again on cement.

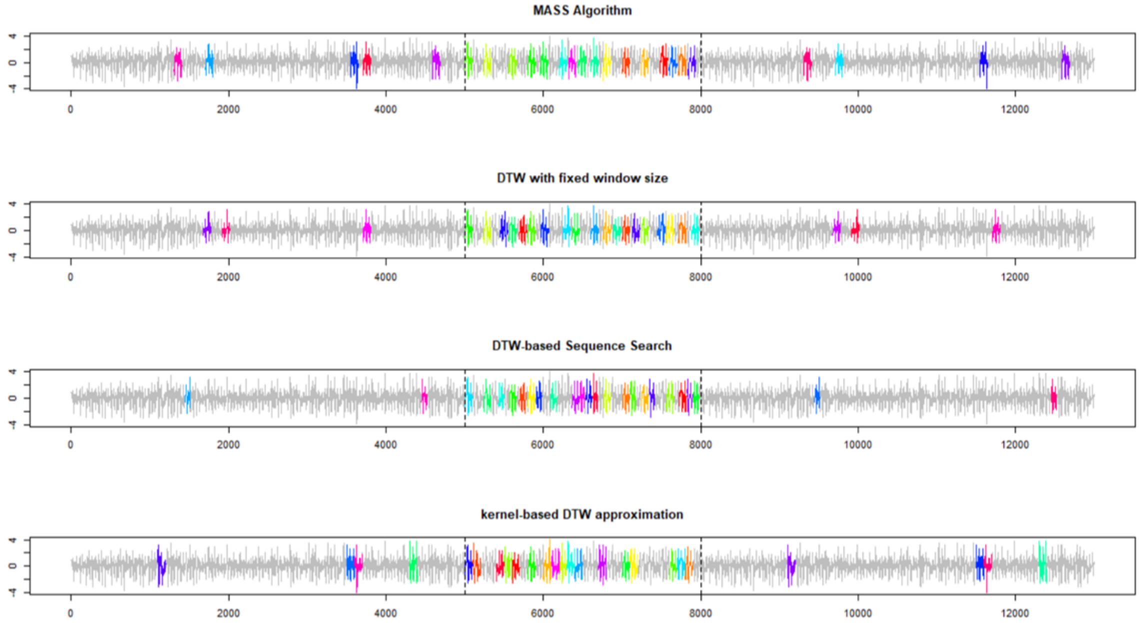

Figure 4. First 25 best matching subsequences identified by four different algorithms on the Experiment 1. Reference pattern consists of 100 acceleration data points collected when a Sony AIBO robot was walking on a carpet. The data stream consists of 5000 data points when the same dog was walking on cement, followed by 3000 walking on a carpet and then again on cement.

5.2. Results on Experiment 2

According to Experiment 1, the DTW-based subsequence search and kernel-based DTW approximations were more effective in identifying those subsequences, which are similar in shape to the reference. Therefore, we have decided to focus Experiment 2 on these approaches. The main differences with Experiment 1 are that, in this case, data streams are multivariate and 22 different references and data streams are considered. Thus, this experiment is devoted to validate the extension of the proposed approach to the multi-references case, summarized in Algorithm 3. The task consists in identifying, for each pair "reference–data stream", the number of matching subsequences whose distance from the specific reference is lower than a given threshold. We named these repetitions "coherent repetitions". Algorithm 2 was applied in two different ways: (a) directly on the multivariate data and (b) on each dimension of the data streams, separately (univariate approach). The sum of the coherent repetitions over the dimensions provides a measure on the quality of the matching between the given reference and data stream. The kernel-based DTW approximation algorithm was applied on the multivariate data streams, only. Figure 5 summarizes the confidence levels between each pair of reference pattern and data stream, respectively for DTW-based sequence search (on the left) and kernel-based DTW approximation (on the right), where each row of the matrix is associated to a reference and each column is associated to the data stream where the reference belongs to. The confidence level is computed as the number of repetitions of the reference within the data stream, in the case of DTW-based subsequence search, and values of the kernel-based distance in the case of the proposed approach. As both represent the same measure of confidence level, we simply rescaled it into $[0, 1]$; more precisely, in the case of the DTW-based subsequence search, from every entry of the matrix, the minimum value on the corresponding row was subtracted, and then the result were divided by the difference between maximum and minimum values on that row. As the kernel-based DTW approximation is by definition in $[0, 1]$, it does not require any further rescaling.

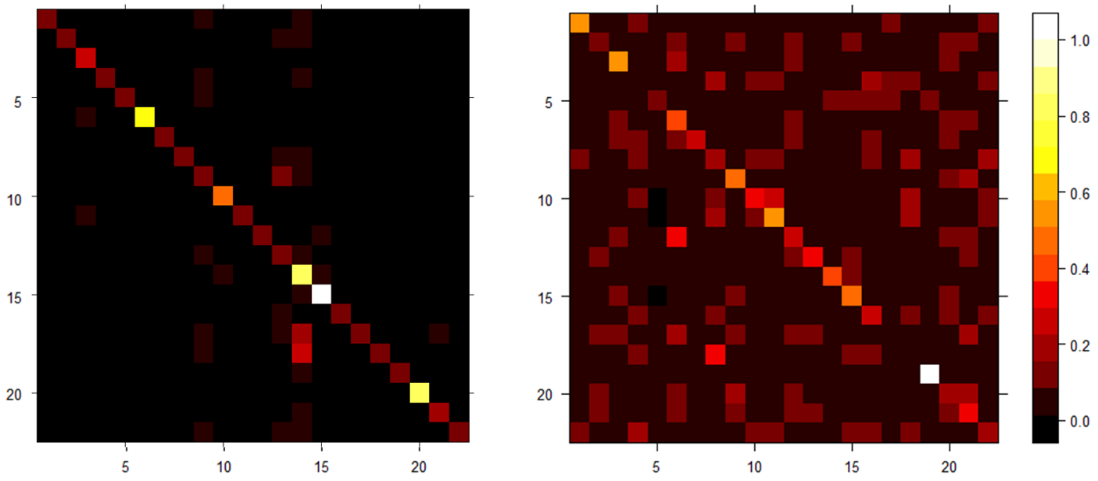

Figure 5. Distance between each pair of reference and data stream, respectively, for DTW-based sequence search (on the **left**) and kernel-based DTW approximation (on the **right**). The brighter the colour the lower the distance.

The DTW-based sequence search was able to correctly associated 19 out of 22 users to their corresponding data streams, whereas our kernel-based DTW approximation algorithm correctly associated 17 out of 22. The proposed kernel-based DTW approximation resulted in a higher error due to its approximating nature; however, this higher error was counterbalanced by a significant reduction in computational time,which is in percentage similar to the previous experiment: 927 s compared to 2056 s required by Algorithm 2. The reduction of the computational cost is slightly lower than the previous case due to the further effort required to optimize the value of the kernel's hyperparameter via Bayesian Optimization.

5.3. Results of Experiment 3

Recall that the correct identification of the five reference patterns was validated by the clinician via visual inspection of the data stream collected during a self-rehabilitation session performed by the subject. Table 1 reports the results for this experiment. In particular, the subsequences identified in the correct portion of the data stream (i.e., "hit") are labeled by "Yes", whereas the "miss" ones are labeled by "No". A label "-" denotes no subsequences identified.

Table 1. DTW-based subsequence search vs. kernel-based DTW approximation for subsequence search.

	Coherent Visually	
Exercise	**DTW-Based Subsequence Search**	**Kernel-Based DTW Approximation**
Flexo-extension of the knee, sit-down position. Five repetitions planned.	Yes Yes Yes No No	Yes Yes Yes Yes Yes
Light squat, stand-up position. Five repetitions planned.	Yes Yes Yes Yes Yes	No No Yes - -
Back extension of the legs, stand-up position. Five repetitions planned.	Yes Yes Yes No No	Yes Yes Yes Yes Yes
Rotate the torso, sit-down position. Five repetitions planned.	Yes Yes Yes Yes No	Yes Yes - - -
Raise and lower the arms, sit-down position. Ten repetitions planned.	Yes Yes Yes Yes Yes No Yes No Yes Yes	Yes Yes Yes Yes Yes Yes Yes Yes - -

From Table 1, it is possible to notice that, except for the second rehabilitation exercise, the traditional DTW-based subsequence search and the proposed kernel-based DTW approximation are quite aligned. This could be due to the basis time-series randomly chosen, which might be not able to provide a good approximation of this specific reference pattern. Even in this case, the reduction in terms of computational costs is similar to previous experiments: ~21 s compared to 37 s required by Algorithm 2.

6. Conclusions

We present an efficient approach of the subsequence search problem in data stream where DTW computation is approximated through kernel learning in the space induced by a feature embedding derived on a set of randomly generated basis time-series.

The validation on three different "small-scale" case studies have been highlighted the potential advantages offered by the proposed approach; basically, a good approximation and a significant lower computational time with respect to a traditional DTW-based implementation. The parallelization schema proposed in Figure 2 makes this approach also applicable to large-scale data streams.

A relevant limitation of the approach is its strong dependence on the initial set of randomly generated basis time-series. Future work should be devoted to define a suitable set of these time-series leading to a robust and effective embedding, starting from the findings provided in [32]. Finally, the relation between the embedding and the (control of) approximation error should be more deeply investigated.

Author Contributions: Conceptualization, A.C. and E.M.; methodology, A.C.; software, Sanislav Fedorov.; validation, S.F., A.C. and E.M.; writing–original draft preparation, A.C. and S.F.; writing–review and editing, E.M. and A.C.; project administration, E.M.; funding acquisition, E.M.

Funding: The research leading to these results has received funding from The Home of Internet of Things (Home IoT) project CUP: E47H16001380009; Call Linea R&S per Aggregazioni; SIDERAB project CUP: E49I18000020009, Project ID 232549; and Call Accordi per la Ricerca e l'Innovazione both under the funding programme POR FESR 2014–2020.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Müller, M. Dynamic time warping. In *Information Retrieval for Music and Motion*; Springer: Berlin, Germany, 2007; pp. 69–84.
2. Han, L.; Gao, C.; Zhang, S.; Li, D.; Sun, Z.; Yang, G.; Li, J.; Zhang, C.; Shao, G. Speech Recognition Algorithm of Substation Inspection Robot Based on Improved DTW. In *International Conference on Intelligent and Interactive Systems and Applications*; Springer: Berlin, Germany, 2018; pp. 47–54.
3. Tang, J.; Cheng, H.; Zhao, Y.; Guo, H. Structured dynamic time warping for continuous hand trajectory gesture recognition. *Pattern Recognit.* **2018**, *80*, 21–31. [CrossRef]
4. Calli, B.; Kimmel, A.; Hang, K.; Bekris, K.; Dollar, A. Path Planning for Within-Hand Manipulation over Learned Representations of Safe States. In Proceedings of the International Symposium on Experimental Robotics (ISER), Buenos Aires, Argentina, 5–8 November 2018.
5. Bauters, K.; Cottyn, J.; Van Landeghem, H. Real time trajectory matching and outlier detection for assembly operator trajectories. In Proceedings of the Internation Simulation Conference, Eurosis, Ponta Delgada, Portugal, 6–8 June 2018.
6. Paiyarom, S.; Tangamchit, P.; Keinprasit, R.; Kayasith, P. Fall detection and activity monitoring system using dynamic time warping for elderly and disabled people. In Proceedings of the 3rd International Convention on Rehabilitation Engineering & Assistive Technology, Singapore, 22–26 April 2009; p. 9.
7. Varatharajan, R.; Manogaran, G.; Priyan, M.K.; Sundarasekar, R. Wearable sensor devices for early detection of Alzheimer disease using dynamic time warping algorithm. *Clust. Comput.* **2018**, *21*, 681–690. [CrossRef]
8. Candelieri, A.; Archetti, F. Global optimization in machine learning: The design of a predictive analytics application. *Soft Comput.* **2019**, *23*, 2969–2977. [CrossRef]
9. Candelieri, A.; Archetti, F. Identifying typical urban water demand patterns for a reliable short-term forecasting–the icewater project approach. *Procedia Eng.* **2014**, *89*, 1004–1012. [CrossRef]
10. Izakian, H.; Pedrycz, W.; Jamal, I. Fuzzy clustering of time series data using dynamic time warping distance. *Eng. Appl. Artif. Intell.* **2015**, *39*, 235–244. [CrossRef]
11. Petitjean, F.; Forestier, G.; Webb, G.I.; Nicholson, A.E.; Chen, Y.; Keogh, E. Dynamic time warping averaging of time series allows faster and more accurate classification. In Proceedings of the 2014 IEEE International Conference on Data Mining, Shenzhen, China, 14–17 December 2014; pp. 470–479.
12. Radoi, A.; Burileanu, C. Retrieval of Similar Evolution Patterns from Satellite Image Time Series. *Appl. Sci.* **2018**, *8*, 2435. [CrossRef]
13. Sakoe, H.; Chiba, S.; Waibel, A.; Lee, K.F. Dynamic programming algorithm optimization for spoken word recognition. *Readings Speech Recognit.* **1990**, *159*, 224.
14. Itakura, F. Minimum prediction residual principle applied to speech recognition. *IEEE Trans. Acoust. Speech Signal Process.* **1975**, *23*, 67–72. [CrossRef]

15. Mueen, A.; Keogh, E. Extracting optimal performance from dynamic time warping. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 2129–2130.
16. Silva, D.F.; Batista, G.E. Speeding up all-pairwise dynamic time warping matrix calculation. In Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, FL, USA, 5–7 May 2016; pp. 837–845.
17. Silva, D.F.; Giusti, R.; Keogh, E.; Batista, G.E. Speeding up similarity search under dynamic time warping by pruning unpromising alignments. *Data Min. Knowl. Discov.* **2018**, *32*, 988–1016. [CrossRef]
18. Keogh, E.J.; Pazzani, M.J. Derivative dynamic time warping. In Proceedings of the 2001 SIAM International Conference on Data Mining, Chicago, IL, USA, 5–7 April 2001; pp. 1–11.
19. Nagendar, G.; Jawahar, C.V. Fast approximate dynamic warping kernels. In Proceedings of the Second ACM IKDD Conference on Data Sciences, Bangalore, India, 18–21 March 2015; pp. 30–38.
20. Cuturi, M.; Vert, J.P.; Birkenes, O.; Matsui, T. A kernel for time series based on global alignments. In Proceedings of the 2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07, Honolulu, HI, USA, 15–20 April 2007; Volume 2, p. II-413.
21. Cuturi, M. Fast global alignment kernels. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), Bellevue, WA, USA, 28 June–2 July 2011; pp. 929–936.
22. Bahlmann, C.; Haasdonk, B.; Burkhardt, H. Online handwriting recognition with support vector machines-a kernel approach. In Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition, Niagara on the Lake, ON, Canada, 6–8 August 2002; pp. 49–54.
23. Sakurai, Y.; Yoshikawa, M.; Faloutsos, C. FTW: Fast similarity search under the time warping distance. In Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, Baltimore, MD, USA, 13–15 June 2005; pp. 326–337.
24. Yen, I.E.H.; Lin, T.W.; Lin, S.D.; Ravikumar, P.K.; Dhillon, I.S. Sparse random feature algorithm as coordinate descent in hilbert space. In *Advances in Neural Information Processing Systems*; The MIT Press: Cambridge, MA, USA, 2014; pp. 2456–2464.
25. Marteau, P.F.; Gibet, S. On recursive edit distance kernels with application to time series classification. *IEEE Trans. Neural Netw. Learn. Syst.* **2014**, *26*, 1121–1133. [CrossRef] [PubMed]
26. Wu, L.; Yen, I.E.H.; Yi, J.; Xu, F.; Lei, Q.; Witbrock, M. Random warping series: A random features method for time-series embedding. *arXiv* **2018**, arXiv:1809.05259.
27. Frazier, P.I. Bayesian Optimization. In *Recent Advances in Optimization and Modeling of Contemporary Problems*; Informs PubsOnLine: Catonsville, MD, USA, 2018; Chapter 11, pp. 255–278, doi:10.1287/educ.2018.0188. [CrossRef]
28. Hutter, F.; Kotthoff, L.; Vanschoren, J. Automatic machine learning: Methods, systems, challenges. In *Challenges in Machine Learning*; Springer: Berlin, Germany, 2019.
29. Feurer, M.; Klein, A.; Eggensperger, K.; Springenberg, J.T.; Blum, M.; Hutter, F. Auto-sklearn: Efficient and Robust Automated Machine Learning. In *Automated Machine Learning*; Springer: Berlin, Germany, 2019; pp. 113–134.
30. Mueen, A.; Zhu, Y.; Yeh, M.; Kamgar, K.; Viswanathan, K.; Gupta, C.; Keogh, E. The Fastest Similarity Search Algorithm for Time Series Subsequences under Euclidean Distance. 2017. Available online: http://www.cs.unm.edu/~mueen/FastestSimilaritySearch.html (accessed on 26 November 2019).
31. Casale, P.; Pujol, O.; Radeva, P. Personalization and user verification in wearable systems using biometric walking patterns. *Pers. Ubiquitous Comput.* **2012**, *16*, 563–580. [CrossRef]
32. Ye, L.; Keogh, E. Time series shapelets: A new primitive for data mining. In Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, 28 June–1 July 2009; pp. 947–956.