Matching incomplete time series with dynamic time warping: an algorithm and an application to post-stroke rehabilitation

Paolo Tormene [1] Toni Giorgino [1,*] Silvana Quaglini Mario Stefanelli

Laboratory for Biomedical Informatics, Computer Engineering and Systems Science, Università di Pavia, Via Ferrata 1, I-27100 Pavia, Italy

Key words: Dynamic programming, Timeseries classification, Nearest neighbour, Motor rehabilitation, Real-time feedback, Post-stroke, Dynamic time warping, Subsequence matching, Wearable sensors
PACS: 05.45.Tp

Summary

Objective: The purpose of this study was to assess the performance of a real-time ("open-end") version of the dynamic time warping (DTW) algorithm for the recognition of motor exercises. Given a possibly incomplete input stream of data and a reference time series, the open-end DTW algorithm computes both the size of the prefix of reference which is best matched by the input, and the dissimilarity between the matched portions. The algorithm was used to provide real-time feedback to neurological patients undergoing motor rehabilitation.

Methods and materials: We acquired a dataset of multivariate time series from a sensorized long-sleeve shirt which contains 29 strain sensors distributed on the upper limb. Seven typical rehabilitation exercises were recorded in several variations, both correctly and incorrectly executed, and at various speeds, totaling a data set of 840 time series. Nearest-neighbour classifiers were built according to the outputs of open-end DTW alignments and their global counterparts on exercise pairs. The classifiers were also tested on well-known public datasets from heterogeneous domains.

* Corresponding author. Tel.: +39 0382 985981. Fax.: +39 0382 526088. E-mail address: toni.giorgino@gmail.com (T. Giorgino)
[1] These authors contributed equally to this work.

Article published in Artificial Intelligence in Medicine Volume 45 Issue 1 (2009) 11-34

Results: Nonparametric tests show that (1) on full time series the two algorithms achieve the same classification accuracy (p-value $= 0.32$); (2) on partial time series, classifiers based on open-end DTW have a far higher accuracy ($\kappa = 0.898$ versus $\kappa = 0.447; p < 10^{-5}$); and (3) the prediction of the matched fraction follows closely the ground truth (root mean square $< 10\%$). Results hold for the motor rehabilitation and the other datasets tested, as well.

Conclusions: The open-end variant of the DTW algorithm is suitable for the classification of truncated quantitative time series, even in the presence of noise. Early recognition and accurate class prediction can be achieved, provided that enough variance is available over the time span of the reference. Therefore, the proposed technique expands the use of DTW to a wider range of applications, such as real-time biofeedback systems.

1 Introduction

Rehabilitation is a challenge in the clinical course of patients affected by an acute neurological event. In particular, rehabilitation after stroke is known to benefit from early start of physical therapy, both in terms of quality of recovery [1,2] and cost-effectiveness [3]. However, obstacles of various nature are met in practice. They range from the difficulty in finding a slot in a rehabilitation structure, to the necessity of motivating the patient – often affected by post-event depression.

Additionally, long waiting lists consume resources: patients that are unable to come back at their home (e.g. because they were living alone) are often kept in the acute ward until a place becomes available in a post-acute rehabilitation structure [4]. This is expensive, because the patient stays in the costly acute ward longer than necessary. There is consequently a motivation for systems that enhance the autonomy of patients in their rehabilitation path.

Current guidelines for stroke prevention and treatment, e.g. the Stroke Prevention and Educational Awareness Diffusion (SPREAD) guideline (cf. recommendations 14.18, 14.25, 15.27 in [5], edition 2007), hint about the potential utility of systems that could foster earlier and prolonged rehabilitation, both in the clinic and at home. The basic idea is to provide patient with self-rehabilitation facilities, able to give them an immediate feedback for rewarding them and/or improving their performance. In the hospital these systems could enhance the traditional rehabilitation practice (which may be limited by lack of personnel and space) by allowing patients to perform more rehabilitation sessions than those regularly scheduled; at home, they could allow therapy to be continued – possibly under remote supervision – implementing the so important continuity of care.

Information technologies are natural candidates to realize rehabilitation aids that can be used locally or with the remote assistance of therapists. In addition to economic and organizational reasons, computerized interventions are appealing because they enable the acquisition of quantitative data on the progress of the therapy (lack of quantification is another well-known problem in the rehabilitation field). A computer-based rehabilitation aid could easily track the therapy and provide a systematic record of its progress.

Rehabilitation after a stroke may involve cognitive training, motor therapy, or both. The neurological rehabilitation system we developed (Product Concept NR) manages the two aspects [6]; this paper focuses on motor therapy and, in particular, on the rehabilitation of the upper limb, because upper limb disability is a frequent outcome of the event.

The paper presents an algorithm that processes body motion signals in order to verify the correct execution of rehabilitation exercises: the repetitions of the upper limb movement have to be counted and classified, evaluating their adherence to a known "correct" reference path, previously recorded with the supervision of a physiotherapist. The problem tackled is to provide feedback to patients in real-time, *before* the exercise is complete and the corresponding time series have been acquired in full. For this purpose, the stream of data from the motion measuring system has to be compared several times per second with suitable references. The output of the algorithm is eventually used to display how close the input is to the proper path.

We want to stress that, while the algorithm is actually embedded into the system for upper limb post-stroke rehabilitation, it is general indeed, and it can be exploited for several kind of template-based time series classification.

2 Movement recognition from the sensorized garment

The algorithm discussed in this paper was designed for detecting arm movements in real-time from a sensorized shirt [6,7]. Motion detection is based on an innovative strain sensor technology, directly printed on garments. Conductive elastomers (CE) are polymer-based materials which exhibit electrical conductivity and can be deposited without significantly altering the mechanical properties of the underlying fabric [8,9]. After vulcanization, the stripes show an electrical resistance around 10-100 kΩ/cm that depends on the printed geometry and varies according to the instantaneous stretch imposed on the fabric. The same material realizes connections to a pocket-sized device (SEW2, Centre Suisse d'Electronique et de Microtechnique) that wirelessly transmits the resistance (strain) measurements to a *patient station* providing feedback to the patient and caregivers in real-time. The current garment prototypes

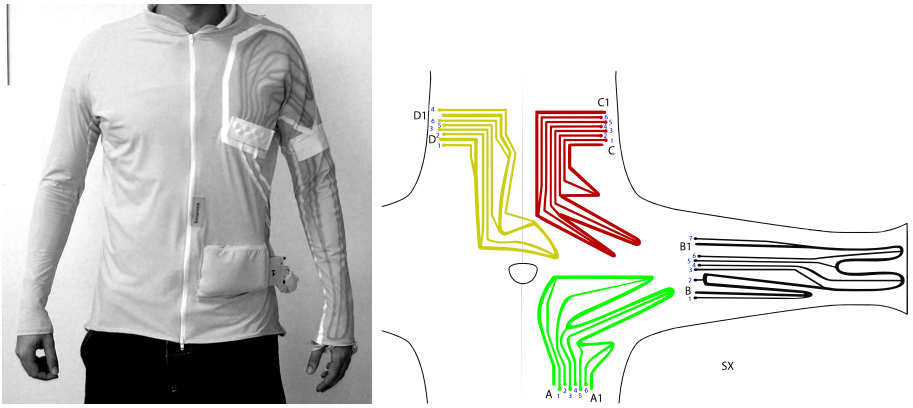

Figure 1. Left: a left-hand sensorized shirt. Right: the layout of strain sensors on the garment.

contain 29 sensing segments distributed over the arm, forearm and shoulders, either left or right depending of the patient's side of hemiplegy (Figure 1). They are sampled in 16-bit precision, approximately 30 times per second. For the sake of brevity, in the following we shall call *time series* the data structure holding 29 simultaneous time-stamped sensor readings for each sampling instant in a given time window.

2.1 Classification problem for strain sensors

The movement recognition problem arises from the fact that there is no known direct correspondence between the amount of fabric stretch of various parts of the garment and biomechanical parameters, e.g. angles between limb segments. Depending on the exercise, sensor readings are in general redundant and highly correlated; the number of sensorized segments is usually larger than, and in no direct correspondence with the degrees of freedom involved. Furthermore, garment-based strain sensors are affected by various sources of uncertainty, which make the computation of limb positions from sensor readings less than straightforward. Finally, measurements are affected by non-linearities of the sensor response to strain, difference in body build, and placement of the garment, even between different sessions for the same subject (Table 1). These difficulties rule out the analytic treatment of the problem.

For this reason, the problem of posture classification from strain readings has been previously tackled by machine learning algorithms [10,11,12]. Such "static" approaches use supervised learning techniques to identify the regions in the input (sensor) space that correspond to relevant key postures in the physical space (e.g. rest position, arm adducted halfway, fully extended limb, typical incorrect postures, and so on). A mapping is established with a calibration procedure that builds a per-patient training set. At run-time, the course of the movement is detected when the feature vector corresponding to

Table 1
Sources of artefacts in garment-based strain sensors

Time scale	**Cause**
$\sim$ ms	Measurement noise
$\sim$ 10 s	Velocity-dependent transients
$\sim 10^2$ s	Sensor drift; displacement over skin
Days	Limited repeatability in wearing
Months	Long term deterioration, wash
Subject	Body size and build

the current garment state traverses the key regions identified at calibration. These machine learning approaches only classify *instantaneous* sensor readings, and therefore fail to take into account the time-dependent behavior of exercises and sensor response. The time scale of velocity-dependent artefacts (1-10 s), for example, overlaps that of body movements under consideration, and recognition accuracy is consequently hampered.

2.2 Pattern-matching approaches

Pattern matching techniques can compare the movement under evaluation with a set of pre-recorded "known" references. This analysis can be a better solution to movement recognition than static approaches, because reference timeseries take into account the time course of the multivariate time series, including potential artefacts. In fact, recognition quality may even be improved by artefacts, on condition that they occur repeatably as a consequence of inputs, because they would add distinguishing features to the signal time series.

Our aim is to check whether a given input stream matches *at most once* a prefix of one of several stored *references*, used as templates. The input stream is sometimes also called *query* or *test timeseries*.

We face two main design choices, namely:

(1) ensuring the analysis can take place in real-time, so that feedback on exercise progress is provided continuously, and not just after the movement is finished (and manually marked as such); and
(2) choosing an algorithm to quantify similarity between time series (goodness of match between the input being evaluated and stored references), irrespective of the pace of execution.

The former requirement rules out those features and preprocessing filters which require the knowledge of future parts of the time series; this set includes e.g. non-local transforms and smoothing by large convolution kernels.

For what concerns the latter, vast amounts of literature explored the classification of time series. The most straightforward approaches for a comparison, e.g. linear time normalization, treat time series as vectors and compute a vector distance; this is clearly ineffective in presence of relative time translations, compressions or expansions. Among the more flexible approaches available for time series analysis, we focus on dynamic time warping (DTW), deferring the a review of other relevant methods to the "Discussion" section.

DTW typically requires that test and reference timeseries are known *in full* to provide a distance, thus violating requirement (1) above. This paper discusses how this constraint can be eliminated.

3 The motor-rehabilitation exercise data set

The reference dataset acquired and analyzed for this study contains 840 time series, each representing one execution of a motor rehabilitation exercise. The set of exercises analyzed includes both deficit-correlated and functional movements, performed both correctly and incorrectly, and at different speeds. Exercises in the dataset were performed by a healthy subject wearing a left-handed sensorized garment as described in the introduction, observing the directions of a physician in a controlled setting. The exercises were performed in sequence according to a pre-defined protocol; immediately after the acquisition, each repetition was labeled with a combination of four integers, $sijk$, whose meaning will be described below. For example, the index j identifies how a given exercise was made (Table 4), and thus can be used to distinguish correct from wrong executions (those with $j = 1 \dots 3$ versus $j = 4 \dots 6$), or to operate finer distinctions. The "incorrect" movements were performed by the same subject, mimicking typical compensatory movements (under supervision). Classification experiments at different granularities will be presented in Sections 5.1.1 to 5.1.3.

The neuro-rehabilitation exercise dataset (called NR in the following) is publicly available [13] and its structure is summarized in Tables 2 – 4. The acquisition protocol consisted of four sessions ($s = 1 \dots 4$); between one session and the next the sensorized garment was taken off, stored for one hour, then worn again. During each session seven types of exercise were executed in sequence ($i = 1 \dots 7$), each in six different modes ($j = 1 \dots 6$). In each mode, the action was repeated five times ($k = 1 \dots 5$). Each element of the dataset, NR_{sijk}, is therefore identified by the four indices specifying the session s, type

Table 2
Overview of the rehabilitation exercise data set used in the classification experiments.

Index	Range	Symbol
Acquisition sessions	$1 \ldots 4$	s
Exercise types per session	$1 \ldots 7$	i
Modes of execution per type	$1 \ldots 6$	j
Repetitions per mode	$1 \ldots 5$	k
Total number of time series	840	NR_{sijk}

Table 3
Description of the seven types of motor rehabilitation exercises analyzed. The description corresponds to the reference (correct) execution.

i	Description of the exercise type
1	Abduction-adduction of the upper limb on a frontal plane
2	Abduction-adduction of the upper limb on a sagittal plane
3	External rotation of the forearm
4	Flexion-extension of the elbow
5	Pronation-supination of the forearm
6	Functional activity: eating
7	Functional activity: combing

of exercise i (Table 3), mode j (Table 4) and repetition k. The acquisition protocol, therefore, foresaw the execution of exercises corresponding to all combinations of the indices, in the order discussed above, for a total count of $4 \times 7 \times 6 \times 5 = 840$ elements.

Each time series is represented by a two-dimensional matrix, in which each row contains 29 simultaneous space-separated readings from strain sensors placed on the sensorized garment. Sampling frequency is approximately 30 Hz and no pre-filtering is applied. For the experiments described in this work, DC offsets are corrected at the beginning of each time series, so that at time one they begin with the null vector, and their leading sections will be aligned regardless of their length.

Since this paper is focused on the dissimilarity computation, we will *not* be concerned with other abstraction (preprocessing, feature extraction) or feature mining steps, which are part of a typical temporal data-mining pipeline [14,15]. Raw time-stamped timeseries will be used in turn as inputs, either in the role of test or reference.

Table 4
Description of the six modes of execution analyzed per exercise.

j	Description	Correctness	Speed
1	Reference	Correct	Slow
2	Reference	Correct	Average
3	Reference	Correct	Fast
4	Movement too small	Incorrect	Average
5	Typical compensatory action (first)	Incorrect	Average
6	Typical compensatory action (second)	Incorrect	Average

4 The dynamic time warping algorithm

The DTW algorithm is based on the concept that the similarity between two time series should be computed by aligning significant patterns; this is done by locally deforming the time axis in order to minimize the cumulative difference between the aligned points. The search space of alignments considered includes all local compressions and shifts; but it can be restricted with easily interpreted constraints, e.g. the set of transitions or *step patterns* allowed at each point of the alignment, as we shall see. Thus, the method is suitable for matching time series containing patterns that are qualitatively similar but have different lengths and paces.

Given a test time series $X = (x_1 \dots x_i \dots x_N)$ and a reference $Y = (y_1 \dots y_j \dots y_M)$, a correspondence between their elements is established through the warping curve $\Phi = (\phi_t, \psi_t)$, with $t = 1 \dots T$. The name stems from the fact that time indices of the two time series will be mapped through the ϕ and ψ functions respectively as we shall see shortly. In the common DTW algorithm the warping curve is subject to a set of constraints, namely:

(1) *start-point constraint*, i.e. the warping curve is anchored at the origin: $\phi_1 = \psi_1 = 1$;
(2) *end-point constraint*, i.e. a global alignment is required, and the mapping covers the time series completely: $\phi_T = N, \psi_T = M$;
(3) *monotonicity*, i.e. no unphysical "time loops" are allowed: $\phi_t \geq \phi_{t-1}$ and $\psi_t \geq \psi_{t-1}$;
(4) *local slope constraints*: only certain relationships, known as step patterns, are allowed between consecutive points in the warping curve (cf. Figure 3, [16], and [17], Table 4.7).

Step patterns list transitions allowed between consecutive matching pairs; for example, they can limit the amount of compression or stretch, i.e. the slope of the warping curve, and allow insertion or deletion of outliers (Figure 3).

Among all the warping curves allowed, the optimal $\hat{\Phi}$ is taken which minimizes the distance between the two warped time series:

$$\hat{\Phi} = (\hat{\phi}_t, \hat{\psi}_t) = \arg\min_{\phi_t, \psi_t} \sum_{t=1}^{T} d(x_{\phi_t}, y_{\psi_t}) m_{t,\Phi} / M_\Phi \tag{1}$$

where $m_{t,\Phi}$ is a local weighting coefficient and M_Φ is a path-dependent normalization such that $M_\Phi = \sum_t m_{t,\Phi}$, and the symbol $d(\cdot,\cdot)$ is an arbitrary non-negative real valued function called *local distance*; its arguments x_{ϕ_t} and y_{ψ_t} are the elements of the warped input timeseries.

It's worthwhile noting that (warped) values of x_i and y_j only enter the computations through the arguments of d. Consequently, X and Y can be multivariate, nominal or mixed, with a suitable choice of d [18]. Therefore, for brevity, in the following we shall omit component indices without loss of generality.

Given an ordered pair of time series, the *minimum cumulative distance* is finally obtained along the optimal mapping $\hat{\Phi}$:

$$D_{\mathrm{DTW}}(X, Y) = \sum_{t=1}^{T} d(x_{\hat{\phi}_t}, y_{\hat{\psi}_t})\, m_{t,\hat{\Phi}} / M_{\hat{\Phi}} \tag{2}$$

Note that the cumulative distance is normalized; as it will be apparent in the following, one usually wants to compare warping curves according to their average *per step* similarity, otherwise short alignments would be unfairly favored. The existence (and value) of a path-independent normalization depends on a suitable choice of step pattern (Section 4.3). We also remark that, in general, D_{DTW} is not symmetric with respect to its arguments, nor does it obey the triangle inequality; nevertheless, for brevity it is customarily called the "DTW distance" between elements X and Y.

The freedom allowed in the choice of input feature space, local distance, step patterns, etc., makes DTW a very flexible alignment approach. Not surprisingly, it has been a popular technique since the 70s, when it was applied to isolated word recognition. Since then, it has been used for clustering and classification in countless domains: biometrics [19,20], process monitoring [21], electrocardiogram analysis [22,23,24], event-related potentials [25], steadiness assessment [26], clustering of gene expression profiles [27,28], just to name a few. The term *time* series may even be misleading, because the warped dimensions can be other than time, e.g. an angle for shape recognition [29,30,31].

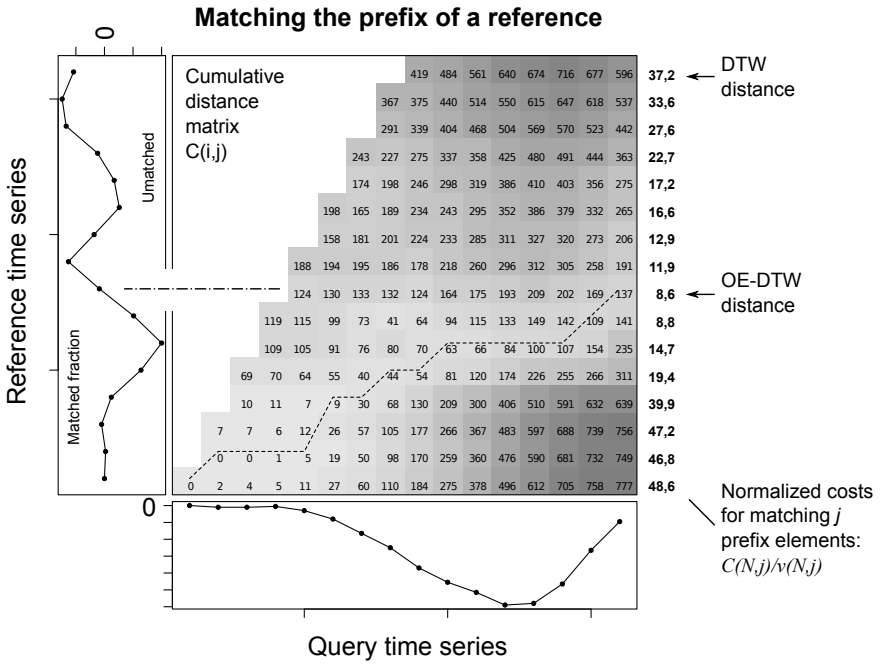

Figure 2. Open-end DTW distance, alignment, and fraction of prefix matched are estimated from the $C(i, j)$ matrix of cumulative costs built from the dynamic programming recursion. For OE-DTW, backtrace starts from the element with minimum *average* cost, computed normalizing the rightmost column. Global DTW alignments pick the solution from the top-right corner.

4.1 Open-end matching

The open-end dynamic time warping (OE-DTW) algorithm discussed in this paper allows the comparison of *incomplete* input time series with *complete* references. It allows, for example, to discover whether a given input matches the first half of one specific reference time series better than the reference as a whole. Furthermore, by iterating OE-DTW over a given *set* of candidate references, one can find which prefix of which candidate best matches the input, as we shall see in Section 4.4.

OE-DTW computes the dissimilarity (or distance) between one input and the best matching forepart of a reference, whereas DTW, being a *global* alignment algorithm, computes the distance between the query and the reference in its entirety. Furthermore, OE-DTW outputs the *fraction* of reference that is matched by the input up to the current time instant (Figure 2).

An intuitive example can illustrate the usefulness of the latter percentage prediction. Focusing on the problem of motion recognition, we assume the digitized version of a full exemplar movement as the reference, and the input stream originating from the motion to be recognized in real-time as a query.

The matching fraction between the two can be interpreted as the fraction of completion of the movement, i.e. whether it is complete or incomplete. The quality of match, or dissimilarity, indicates how likely is that the movement performed is explained by the reference being considered (the choice of the reference set is discussed in more detail in Section 6.1).

By iterating the prediction multiple times incrementally, it is possible to obtain a feedback about the progress of what we are observing, which is very useful e.g. in a real-time feedback application. As we shall discuss in Section 6.6, an incremental update for additional input elements only incurs a constant computational cost.

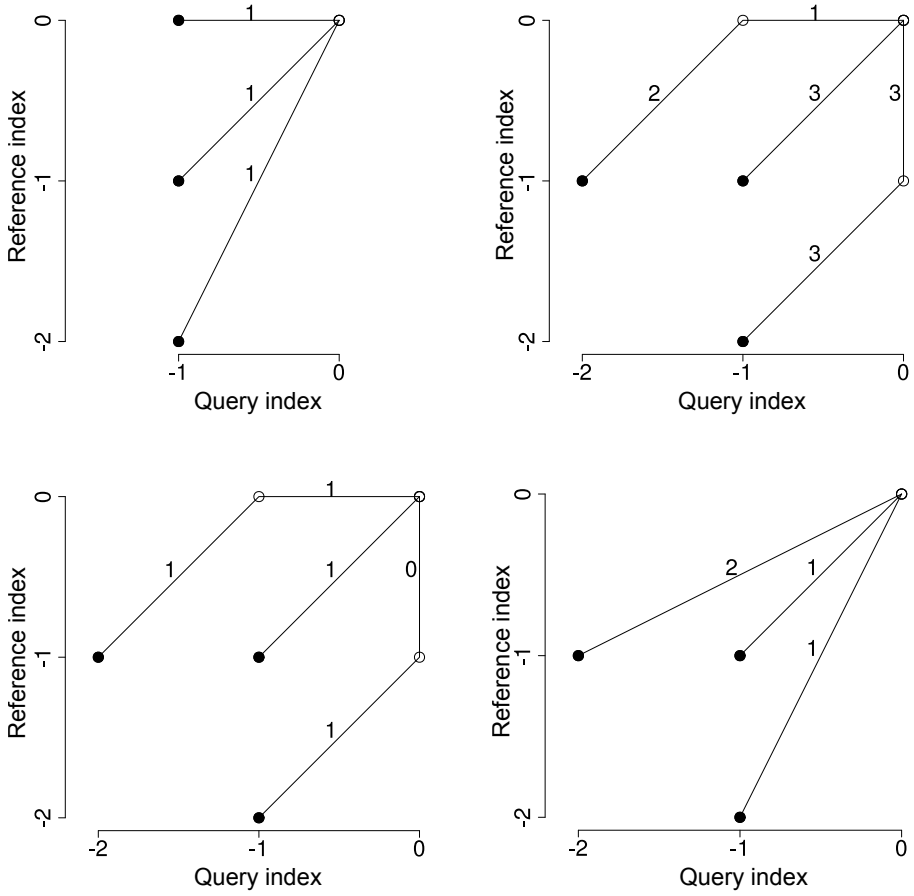

Figure 3. Top left: the asymmetric recursion (6) with the three allowed transitions and corresponding weights; the local slope of the warping curve is limited between 0 and 2. Top right: a different step pattern used e.g. in Mori et al. [32]. Bottom: Rabiner-Juang's [17] unsmoothed type II (c) and type III (c).

4.2 The open-end algorithm

The basic idea of the open-end DTW algorithm is, given one input X and one reference Y, to compute several DTW distances between X (taken as a whole) and multiple references $Y^{(j)}$ truncated at *all* possible points $j = 1 \dots M$. Among the results, one would choose the one with the least DTW distance from the input. Let $Y^{(j)}$ be a time series containing all the j leading elements in Y, i.e. $Y^{(j)} = (y_1, \dots, y_j)$. One therefore defines

$$D_{\mathrm{OE}}(X, Y) = \min_{j=1\dots M} D_{\mathrm{DTW}}(X, Y^{(j)}). \tag{3}$$

Remarkably, a solution to (3) can be computed very efficiently, as we shall see shortly.

The first steps of DTW and OE-DTW are similar. Both are based on the construction of a matrix of cumulative costs C. First, given time series X and Y as above, a *local distance* matrix $L(N \times M)$ is initialized with the distance

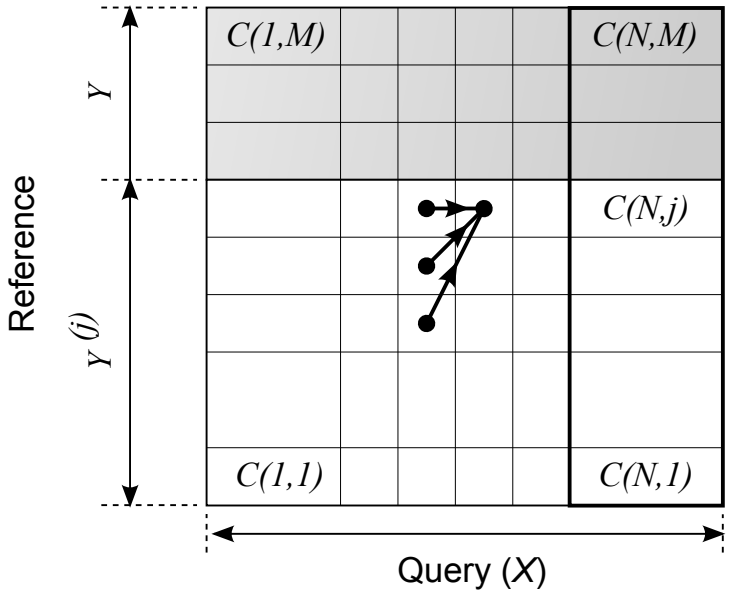

Figure 4. If the reference Y were truncated at j, elements in grey would disappear, but the the remaining elements would be unaffected. This implies that the rightmost column, $C(N, j)$, holds the (unnormalized) DTW distance beween time series X and $Y^{(j)}$ (Y truncated at j). The asymmetric step pattern is shown, but the relation holds in general.

between test elements at time $i = 1 \dots N$ and reference elements $j = 1 \dots M$:

$$L(i, j) = d(x_i, y_j). \tag{4}$$

Second, the *cumulative distance* matrix $C(N \times M)$ is computed from L through an iterative formula which depends on the step pattern chosen. $C(i, j)$ represents the minimum sum of local distances along the alignment paths leaving from $(1, 1)$ and pairing the i-th sample in the input with the j-th sample in the reference.

Among the transition sets available, for clarity we exemplify OE-DTW with a simple asymmetric step pattern (top left of Figure 3). The recursion to compute C is accordingly [16]:

$$C(1, 1) = L(1, 1); \tag{5}$$

$$\begin{aligned} C(i, j) = \min \{ & \; C(i-1, j), \\ & \; C(i-1, j-1), \\ & \; C(i-1, j-2) \} + L(i, j); \end{aligned} \tag{6}$$

where i and j are the test and reference sequence indices respectively. Uninitialized elements and invalid indices on the right hand side should be ignored. We refer the reader to Chapter 4 in Rabiner and Juang's Book [17] for a categorization of patterns and the corresponding iterative formulas.

DTW and OE-DTW differ in how the final distance is computed. The common DTW algorithm always takes the cumulative distance from element $C(N, M) \sim D_{\mathrm{DTW}}(X, Y)$, while OE-DTW compares elements in the last column of $C(N, j)$, $j = 1 \dots M$ (Figure 2). In fact, once the matrix C is computed, all of the partial costs required on the right-hand side of (3) are available at once. In other words, one finds in $C(N, j), j = 1 \dots M$, orderly, the global cost of the best alignment that maps the full input into the elements $1 \dots j$ of the reference timeseries [33]. The property is apparent from the recursion formula, and because the monotonicity constraint imposed on ϕ implies that template elements y_k with $k > j$ bear no influence on $C(N, j)$ (Figure 4).

The optimal OE-DTW alignment is obtained from the rightmost [2] column of C, divided by the corresponding normalization factor $\nu(N, j)$ (discussed in the next section). The open-end distance between the two time series is therefore computed as follows:

[2] We use plot-like indexing: the first index grows rightwards in rows and the second upwards in columns.

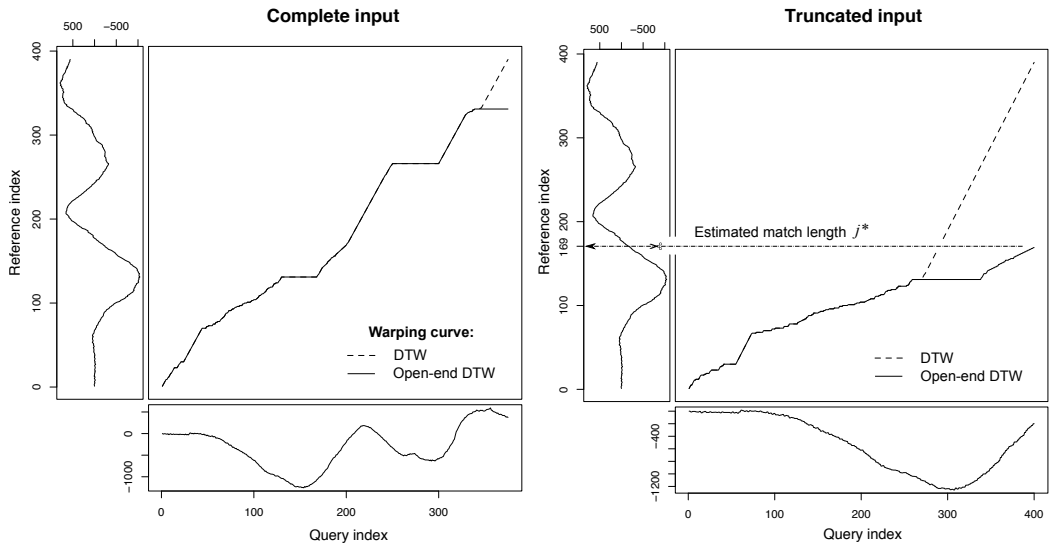

Figure 5. DTW (dashed) and open-end DTW (solid) warping curves compared. Left: if the whole query is available, the two solutions mostly overlap. Right: if only a prefix is available (in the example, around 50%), OE-DTW provides a distance, a meaningful alignment, and estimates the fraction of the reference matched.

$$
\begin{aligned}
j^*(X,Y) &= \arg\min_{j=1\ldots M} \frac{C(N,j)}{\nu(N,j)} \\
D_{\mathrm{OE}}(X,Y) &= \frac{C(N,j^*)}{\nu(N,j^*)} \\
f_{\mathrm{OE}}(X,Y) &= 100\, j^*/M,
\end{aligned}
\tag{7}
$$

where D_{OE} is the estimated distance and f_{OE} is the fraction of reference matched (normalized to 100 for clarity). They are the outputs of the OE-DTW algorithm. The warping curve (ϕ_t, ψ_t) can be obtained, if desired, back-tracing from element (N, j^*) through $(1, 1)$.

Figure 5 compares OE-DTW versus DTW alignments computed for a complete input (left) and a prefix (right); examples are taken from elements NR_{1111} and NR_{1112} of the NR dataset (in the latter, each element was repeated twice to allow non-trivial DTW alignments; only the fourth component out of the 29 is considered for clarity).

4.3 Normalization

In equation (3) and, later, for classification purposes, we align one given input with several reference timeseries to pick the one which best fits the query. It is therefore important that the distance value can be properly normalized with respect to the length of the match, so that a long match is preferred over a short one, if the former has a better alignment *on average*.

Table 5
Normalization functions for well-known weighting types; OE-DTW can be used with step patterns in all three groups. R-J types follow Rabiner's slope weight classification (cf. [34], Figure 7); "Symmetry" refers to the symmetric/asymmetric forms of dynamic programming introduced by Sakoe-Chiba (cf. Table I in [16]). Type (c′) is similar to R-J type (c) with test and reference interchanged, normalized along the reference length.

Slope weighting	Symmetry	$\nu(N, j)$
R-J type (c)	Asymmetric	N
Type (c′)	Asymmetric	j
R-J type (d)	Symmetric	$N + j$

For the recursion formula (6) to be an exact solution to the minimization problem (1), the normalization constant M_Φ should depend only on the path endpoints, i.e. on the length of the input and the length of reference matched; it should not depend on the specific shape of the warping path. In other words, a function ν should exist such that $M_\Phi = \nu(N, M)$ for all Φ considered [17]. Some step pattern categories which satisfy this criterion, and can therefore be used with OE-DTW, are listed in Table 5. For instance, the asymmetric step pattern (6) constrains the alignment to consume each element of the test exactly once, and therefore the warping curve will have as many elements as the input sequence; the normalization function would therefore be the constant $\nu(N, j) = N$.

The experiments shown in this paper have been performed with a type (c) step pattern. Type (c) and (d) step patterns have advantages over a type (c′) used e.g. in Mori et al. [32], Wu et al. [35], which are discussed in Section 6.7.

4.4 Classification

For an application of OE-DTW, we now focus our attention on the supervised classification of a possibly incomplete input query X; the part of it which is available will be compared with P previously stored, complete, reference templates Y_p, $p = 1 \dots P$. Assuming that C classes are defined, we denote the (known) class of Y_p as $c(p) \in \{1 \dots C\}$.

The OE-DTW algorithm, as discussed above, can be used as a dissimilarity measure between the input query and the best matching prefix of each reference. A simple solution to classify X is therefore to use $D_{OE}(X, Y_p)$ for a Nearest Neighbor (1-NN) classifier. The algorithm predicts that the incomplete

query belongs to the same class c^* as its "closest" (most similar) template:

$$c^*(X) = c\left(\arg\min_p D_{OE}(X, Y_p)\right).$$

5 Results

We ran classification experiments on publicly available datasets: first, on the motor rehabilitation dataset NR, for the task which motivated the research [13] (section 5.1); second, on the public UCR repository [36], containing time series from heterogeneous domains (section 5.2). The experiments used the type (c) recursion of eq. (6) for the reasons discussed in section 6.2.

5.1 Classification of motor rehabilitation exercises

This section presents three classification problems on which we tested the performance of 1-NN classifiers based on DTW and on its open-end counterpart. The three experiments divided the NR dataset respectively into: two classes of movements (correct versus incorrect, Section 5.1.1), four classes (correct versus different types of error, Section 5.1.2) or six classes (discerning between all the defined execution modes, including the recognition of the speed of correct movements, Section 5.1.3). The latter experiment has been set up as reference, being aware that DTW should recognize patterns regardless of their pace, as we shall see in the discussion.

We solved the classification problems in four conditions: first, we tested an OE-DTW based 1-NN classifier for the recognition of truncated inputs, comparing the first half of each query with the whole length of each reference; second, we used the same algorithm to classify complete inputs. For a baseline, we repeated the two tests using a traditional DTW-based 1-NN classifier.

Note that the objective of all the previously described experiments is not to recognize *which* exercise type (index i in Table 2) is being performed, but to classify *how* it is executed (index j): we assume to know which exercise is being performed, and we want to automatically recognize the quality of the movement. This implies that we divide the set of 840 time series into seven subsets ($i = 1 \dots 7$) of 120 elements each, and iterate the experiments on these subsets. We quantify the performance of classifiers using leave-one-out cross-validation, i.e. classifying each of the 120 testing time series using the other 119 as references.

Table 6
Performance of DTW and OE-DTW on three problems in movement classification (recognition of 2, 3 or 6 classes). Results resumed as "min (mean) max" for the seven exercise types.

		2 classes			*4 classes*	*6 classes*
Input	**Alg.**	**Accuracy (%)**	**Sensitivity (%)**	**Specificity (%)**	**Accuracy (%)**	**Accuracy (%)**
Complete	DTW	97.5 (99.2) 100.0	95.0 (98.3) 100.0	100.0 (100.0) 100.0	95.8 (97.9) 100.0	84.2 (87.7) 93.3
Complete	OE	95.0 (98.7) 100.0	90.0 (97.4) 100.0	100.0 (100.0) 100.0	92.5 (97.4) 100.0	84.2 (87.3) 93.3
Half	DTW	63.3 (77.1) 92.5	71.7 (83.1) 98.3	45.0 (71.2) 88.3	41.7 (61.9) 78.3	20.0 (27.4) 39.2
Half	OE	93.3 (95.2) 98.3	86.7 (91.0) 96.7	98.3 (99.5) 100.0	90.0 (93.3) 97.5	79.2 (83.6) 91.7

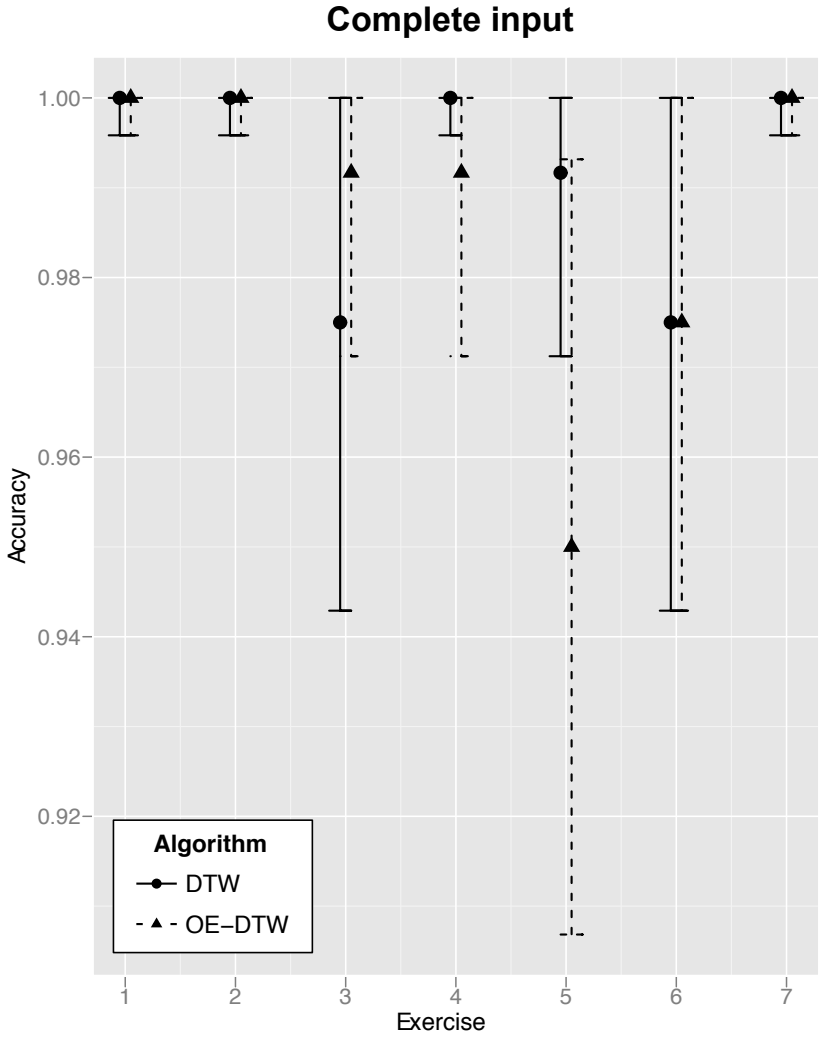

Figure 6. Accuracy of DTW and OE-DTW on the classification of *complete* input queries for the 2-class problem (correct/incorrect). Results are grouped by exercise types. The error bars show the textbook 95% confidence interval.

5.1.1 *Prediction of execution correctness (2-class problem)*

The first experiment had the aim to discern whether an exercise under analysis had been correctly or incorrectly performed. The original purpose of the rehabilitation feedback was in fact to estimate the correctness of exercises performed by patients without direct medical supervision. The class of *correct* movements includes the time series having index $j = 1, 2, 3$, while *incorrect* movements have index $j = 4, 5, 6$ (Table 4).

On the left part of Table 6 we can read the performance of the compared classifiers. On full inputs, DTW and OE-DTW achieved about the same mean accuracy (99.2% for DTW, versus 98.7% for OE), sensitivity (98.3% versus 97.4%) and both had perfect specificity. On truncated inputs, the OE variant

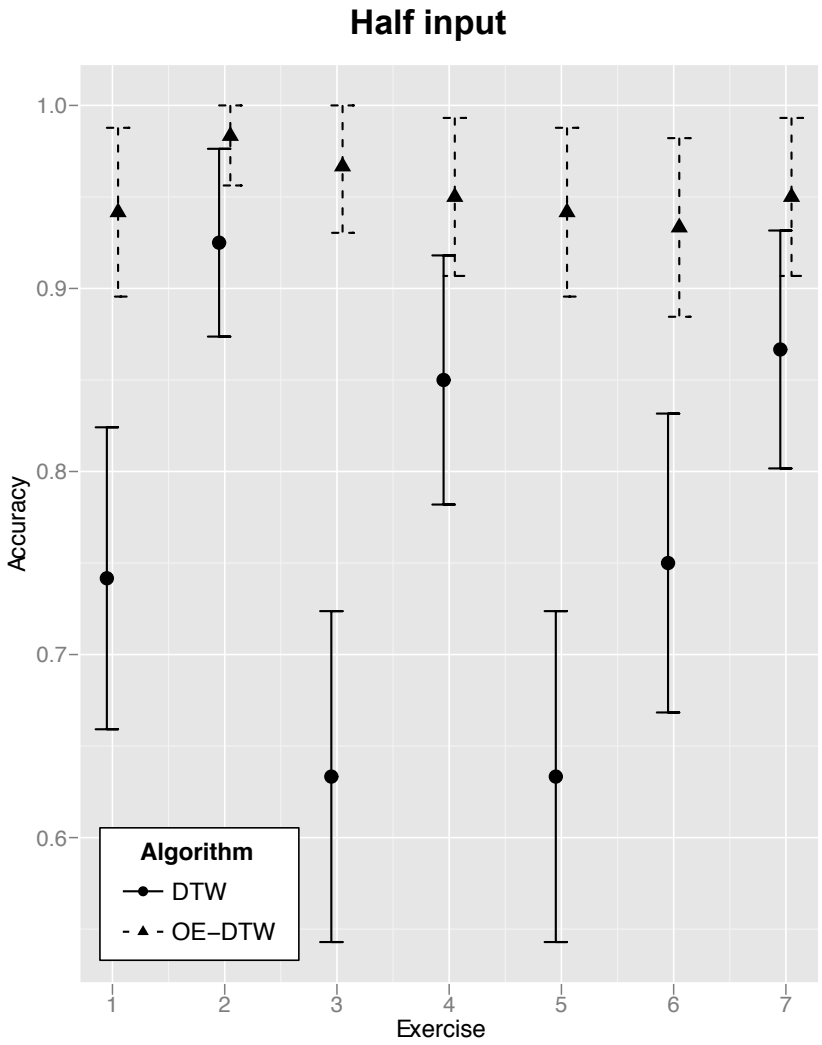

Figure 7. Accuracy of DTW and OE-DTW based 1-NN classifiers, on the problem of 2-class (correct/incorrect) classification of *truncated* (half) queries. Results are grouped by exercise types. The error bars show the textbook 95% confidence interval.

outperformed DTW, as expected (mean accuracy 77.1%, versus 95.2%; mean sensitivities 83.1% versus 91.0%; mean specificities 71.2% versus 99.5%).

Figures 6 and 7 present the accuracies of these methods separately for each of the seven exercise types analyzed. The error bars represent the 95% confidence interval computed using the *textbook limit* [37]:

$$\tau = \epsilon \pm \left(0.5/M + 1.96\sqrt{\epsilon(1-\epsilon)/M}\right)$$

where τ is the estimated accuracy, ϵ is the mean accuracy obtained in the tests and M is the count of predictions. For a proper statistical inference on the relative performance of DTW with respect to OE-DTW in the global and truncated cases, we defer the discussion to Section 5.2, where more sound conclusions can be drawn on the basis of 20 independent data sets.

Looking at the plots, we anticipate that the OE-DTW algorithm generalizes DTW, since the two methods almost overlap in the standard case in which input time series are complete, but only the OE variant gives acceptable results on truncated inputs.

5.1.2 Recognition of correctness vs. error type (4-class problem)

The second experiment had the objective to recognize if the analyzed movement was correctly performed and, if not, to predict the specific error. As in the 2-class problem, the class of correct movements includes the time series having index $j = 1, 2, 3$, whereas modes $j = 4, 5, 6$ are considered each as a separate class.

This analysis provides information which could be very useful in the medical application object of this work. Let us suppose, for instance, that a patient is practicing using a sensorized garment, and that a correct/incorrect classifier (Section 5.1.1) recognizes some incorrect executions. A remote feedback system would communicate this information to a medical therapist, who would be then aware that the patient is not properly doing the prescribed therapy, but with no details about *what* was the mistake (e.g. a typical compensatory movement, or if the movement was almost correct, but too small). A 4-class classifier could instead return more detailed information to the physician, who could consequently plan a targeted intervention.

The results, resumed in Table 6, are similar to those obtained for the 2-class problem. The mean accuracies obtained by DTW are 97.9% for full inputs and 61.9% for truncated inputs; OE achieves almost the same mean accuracy for full inputs (97.4%) and, again, far better 93.3% for truncated inputs.

Table 7
Comparison of the κ-statistic for DTW and OE-DTW for the three problems of classification analyzed (recognition of 2, 3 or 6 classes). Results resumed as "min (mean) max" for the seven exercises.

		2 classes	*4 classes*	*6 classes*
Input	**Alg.**	**κ statistic**	**κ statistic**	**κ statistic**
Complete	DTW	0.950 (0.983) 1.000	0.937 (0.968) 1.000	0.810 (0.853) 0.920
Complete	OE	0.900 (0.974) 1.000	0.885 (0.960) 1.000	0.810 (0.847) 0.920
Half	DTW	0.267 (0.543) 0.850	0.198 (0.447) 0.682	0.040 (0.129) 0.270
Half	OE	0.867 (0.905) 0.967	0.845 (0.898) 0.962	0.750 (0.803) 0.900

5.1.3 Prediction of execution mode (6-class classification)

The aim of the third experiment is to separate in different classes each of the six modes of execution. Considering that most of the variability between movements of modes $j = 1, 2, 3$ lies in their different speed, one would expect that a DTW-based 1-NN classifier should *not* be able to discern between them. Interestingly, on the NR dataset it turns out that this is not the case. This may be explained by velocity-dependent artefacts [38]: fast movements are more affected by non-linear effects with respect to similar, slower, movements, as we shall discuss later.

As in the previous two problems, accuracies are resumed in Table 6, and can be similarly interpreted. Again, the mean accuracies of the algorithms are almost the same (87.7% versus 87.3%) in the case of full inputs, and very different (27.4% versus 83.6%) in the case of truncated inputs.

5.1.4 Comparison between problems

For a comparison between the three problems, we account for different probabilities of random agreement by comparing Cohen's κ-statistic [39,40]:

$$\kappa = \frac{\pi_1 - \pi_2}{1 - \pi_2}$$

where π_1 is the accuracy obtained by the classifier under test and π_2 is the proportion of elements that would be guessed correctly by a random classifier on the same problem; $\kappa = 0$ means that a classifiers is as good as a random one; $\kappa = 1$ is achieved by "perfect" classifiers which never fail their predictions.

Table 7 compares κ-statistics of the two algorithms on the three problems defined (2-4-6 classes). The table reports minimum, mean and maximum values for the seven exercise types, while all of the seven values are shown in Figure 8. From both the table and the plot it is apparent that κ for the 6-class problems is noticeably lower than the others; the 2-class and 4-class problems have similar performance. This was expected, as introduced in Section 5.1.3,

and it is a measure of the difficulty to discern between modes $j = 1, 2, 3$, which represent movements performed at different speeds but otherwise very similar to each other. This separation is however smaller than what one would expect from "perfect" time-independent sensors, as we explain with an example. Let us assume that a 6-class classifier perfectly recognizes modes $j = 4, 5, 6$ and randomly labels modes $j = 1, 2, 3$, which only differ by time stretches. The accuracy of this classifier would be $\pi_1 = 2/3$; since $\pi_2 = 1/6$, κ would be 0.6, far lower than what is observed (for OE-DTW on truncated queries minimum κ was 0.750 and mean was 0.803). The differences may be explained by the non-linearity of the strain-sensing technology used; the faster the movements, the more the non-linear response becomes important [12,41].

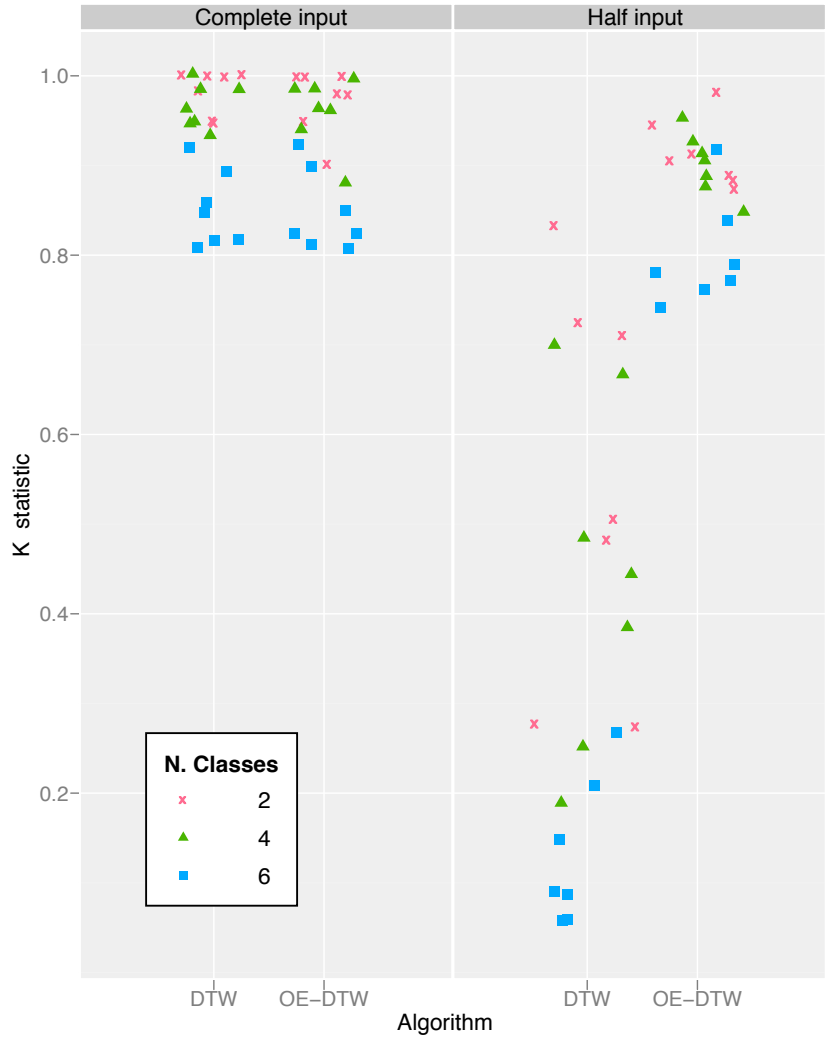

Figure 8. Performance of DTW and OE-DTW on the three classification problems (2, 4, or 6 classes). Seven points are shown for each algorithm, showing corresponding Cohen's κ for the seven exercise types.

5.2 Classification of datasets from heterogeneous domains

Although OE-DTW has been developed for a specific real-time movement recognition problem, for generality we have also evaluated it on the datasets in the publicly available UCR repository [36]. At the time of writing, the UCR repository comprises 20 sets collected from a variety of domains. Each dataset contains a different number of single-variate time series (Table 8). All of the timeseries are pre-labeled with a nominal class identifier, the number of classes varying between 2 and 50 per dataset. The procedures followed for collection, labelling, and the interpretation of the classes depend on the specific dataset; details for each are found in the documentation and references accompanying the repository. In the experiments we honored the proposed splits into training and testing sets.

We arranged a set of experiments to test the following hypotheses:

(1) when we align *part* of a test time series from the testing set with complete references from the training set, an 1-NN classifier based on the OE-DTW distance *outperforms* an analogous classifier based on the DTW distance (Section 5.2.1);
(2) when the considered time series are *complete*, the OE-DTW and DTW classifiers have *the same* performance (Section 5.2.2);
(3) the percentage prediction is accurate (Section 5.3).

The time series were re-sampled to be at most 100 elements long before each alignment; shorter time series stayed unmodified. This step has been taken in order to reduce computational costs; it may however cause loss of information in the high frequency domain (especially important e.g. in the *Lightning* datasets). The statistical tests have been computed with the R software [42], assuming $\alpha = 0.05$ as common.

5.2.1 OE-DTW versus DTW for partial matches

First, we show that OE-DTW performs well in its intended setting, i.e. on partial time series. For each data set, we take the leading part of each test instance and show that they are better labeled – on average – by an OE-DTW-based 1-NN classifier rather than by an analogous DTW-based classifier, taken as a baseline. For this experiment we kept the first half ($f_{\text{true}} = 50\%$) of each test time series, discarding the rest. This fraction is arbitrary; OE-DTW can accurately predict the class as long as enough information is contained in the leading part of the sequence. Results are reported in Table 8 under the column "Half input".

We compared the classifiers with a nonparametric two-sided paired Wilcoxon

Table 8
Accuracies of DTW- and OE-DTW-based 1-NN classifiers on the 20 UCR datasets; C is the number of classes. Column "Half input" lists results obtained when only the first half ($f_{\text{true}} = 50$) of each test time series is aligned with all available references. This is the intended OE-DTW setting, discussed in Section 5.2.1. Column "Complete input" shows results on unabridged time series ($f_{\text{true}} = 100$, Section 5.2.2). OE-DTW clearly outperforms reference DTW on partial time series, and is not worse on complete time series.

					Accuracy of 1-NN classifier			
			Set size		Half input		Complete input	
	Data set	C	train	test	DTW	**OE**	DTW	OE
1	50words	50	450	455	0.066	0.490	0.776	0.712
2	Adiac	37	390	391	0.049	0.371	0.573	0.570
3	Beef	5	30	30	0.47	0.50	0.50	0.47
4	CBF	3	30	900	0.634	0.989	0.998	0.999
5	Coffee	2	28	28	0.54	0.57	0.75	0.75
6	ECG200	2	100	100	0.36	0.84	0.82	0.84
7	FaceAll	14	560	1690	0.1308	0.8130	0.7675	0.8467
8	FaceFour	4	24	88	0.55	0.88	0.85	0.88
9	fish	7	175	175	0.120	0.611	0.737	0.737
10	Gun_Point	2	50	150	0.673	0.813	0.900	0.900
11	Lighting2	2	60	61	0.67	0.72	0.80	0.72
12	Lighting7	7	70	73	0.30	0.56	0.78	0.71
13	OliveOil	4	30	30	0.70	0.87	0.87	0.87
14	OSULeaf	6	200	242	0.145	0.492	0.636	0.665
15	SwedishLeaf	15	500	625	0.072	0.648	0.789	0.779
16	synthetic_control	6	300	300	0.167	0.973	0.987	0.987
17	Trace	4	100	100	0.56	0.72	0.99	0.96
18	Two_Patterns	4	1000	4000	0.2460	0.5310	1.0000	0.9940
19	wafer	2	1000	6164	0.4262	0.9797	0.9838	0.9851
20	yoga	2	300	3000	0.5063	0.7877	0.8317	0.8260

Signed-Ranks test, as suggested by Demšar [43], testing the null hypothesis that the two have the same classification accuracy. The test yields a p-value of $1.907 \cdot 10^{-6}$, and therefore the null hypothesis can be rejected; the accuracies of OE-DTW are greater than DTW on all datasets and thus we affirm that

OE-DTW outperforms DTW on the recognition of partial time series.

An alternative approach for comparing classifiers, suggested by Salzberg [44], is to perform binomial tests. We counted how many test time series were correctly labeled by the OE-DTW-based classifier only (A), by the DTW-based classifier only (B), by both (C), and how many were incorrectly classified by both (D). Ties ($C + D$) are discarded, and binomial tests are performed for each dataset for the null hypothesis that A and B are equally probable. For most of the datasets we obtained a p-values less than 0.0026 (i.e. α with a conservative Bonferroni adjustment for 20 tests), which confirms that the null hypothesis can be rejected.

5.2.2 OE-DTW versus DTW for global alignments

An important feature of OE-DTW is that it still works satisfactorily when applied to complete time series, i.e. in the traditional context of DTW. We therefore repeated the DTW versus OE-DTW experiments of the previous section, this time retaining test instances un-truncated.

Accuracies of the two classifiers on the 20 given datasets are reported in Table 8 under column "Full input"; they seem very close, suggesting that there is no difference in accuracy between the two classifiers. A paired two-sided Wilcoxon Signed-Ranks test yields a p-value = 0.32, which is compatible with the null hypothesis that accuracies are the same.

Counting agreements and disagreements between the predictions, 98% of the examples were labeled equally by the two algorithms. The number of discordant predictions is very small with respect to the agreements, and therefore, as noted by Salzberg [44, Sec. 3.1], the binomial test may not be significant. However, tests on all but three datasets (1, 7 and 18) show no clue against the null hypothesis that OE-DTW has the same performance as DTW even when used in its proper context (DTW beats OE-DTW on datasets 1 and 18; OE-DTW beats DTW on dataset 7). We conclude that DTW and OE-DTW have similar performance on global alignments.

5.3 Reliability of match length prediction

Once a test instance is classified via OE-DTW, the next point to show is that the percentage reported by OE-DTW is accurate, i.e. it reflects the amount of reference truly matched by the query. To quantify this prediction we arranged an experiment comparing artificially truncated time series with complete ones known to belong to the same class. The experiment was run both on the NR and UCR datasets.

5.3.1 Match length prediction for motor rehabilitation exercises

The rationale behind this experiment was to form pairs of similar time series; one of the members of the pair is truncated at a known fraction f_{true}, and aligned to the other via OE-DTW. The aim is to verify if the predicted f_{OE} is close to the known input length. To ensure that the timeseries of each pair are actually similar, they are taken from the same session, exercise and mode.

Formally, we split the NR data set in 24 subgroups, homogeneous in s, i, j. Five elements, $k = 1 \dots 5$, belong to each subgroup, from which 20 ordered timeseries pairs (k, k') with $k \neq k'$ are formed. For each pair, we compute 50 open-end alignments truncating the input at a fraction f_{true}, iterating $f_{\text{true}} = 2 \dots 100\%$ at 2% increments, and finally computing the corresponding f_{OE}:

$$f_{\text{OE}}(f_{\text{true}}; s, i, j, k, k') = f_{\text{OE}}(\text{NR}^{(f)}_{sijk}, \text{NR}_{sijk'})$$

where $f = \lfloor f_{\text{true}} \cdot \text{length}(\text{NR}_{sijk}) \rfloor$ is the number of input elements within f_{true}, rounded to integer. We therefore collect one predicted length per each combination of f_{true}, couple, and subgroup tested ($50 \times 20 \times 24$ values).

Figure 9 shows the distribution of f_{OE} with respect to f_{true}, grouped by exercise type i (quartile bands with median dots). As expected, the median values are approximately on the diagonal. The root mean square (RMS) of residuals between f_{OE} and f_{true}, averaged over all indices but i, are shown in Table 9 (top row). RMS of differences between the medians of f_{OE} for each f_{true} (black dots) and f_{true} itself are also given (bottom row). We conclude that match length prediction is accurate within a few percent, with the expected exception of exercise type $i = 5$ (that is also poorly classified).

As noted above, we never align time series with themselves but with those that, being in the same class and session, are supposedly most similar; consequently, f_{true} is not a ground truth but an approximation. In fact, two time series belonging to the same subgroup may still have misalignments due to the variability within each class (e.g. composite motor exercises which consist of a sequence of sub-movements spaced out by pauses of variable length); the match between corresponding instants of such time series can therefore not be expected to be exactly linear. However, since for each test-reference couple we also observe the corresponding pair with reversed roles, misalignments partially average out.

Note that the plots in Figure 9 and 10 do not depict average warping curves because, for each f_{true}, the future elements of the input are not available, so the prefix of the warping curve is optimal only up to the considered time instant; it may change when subsequent fractions of the input stream become known.

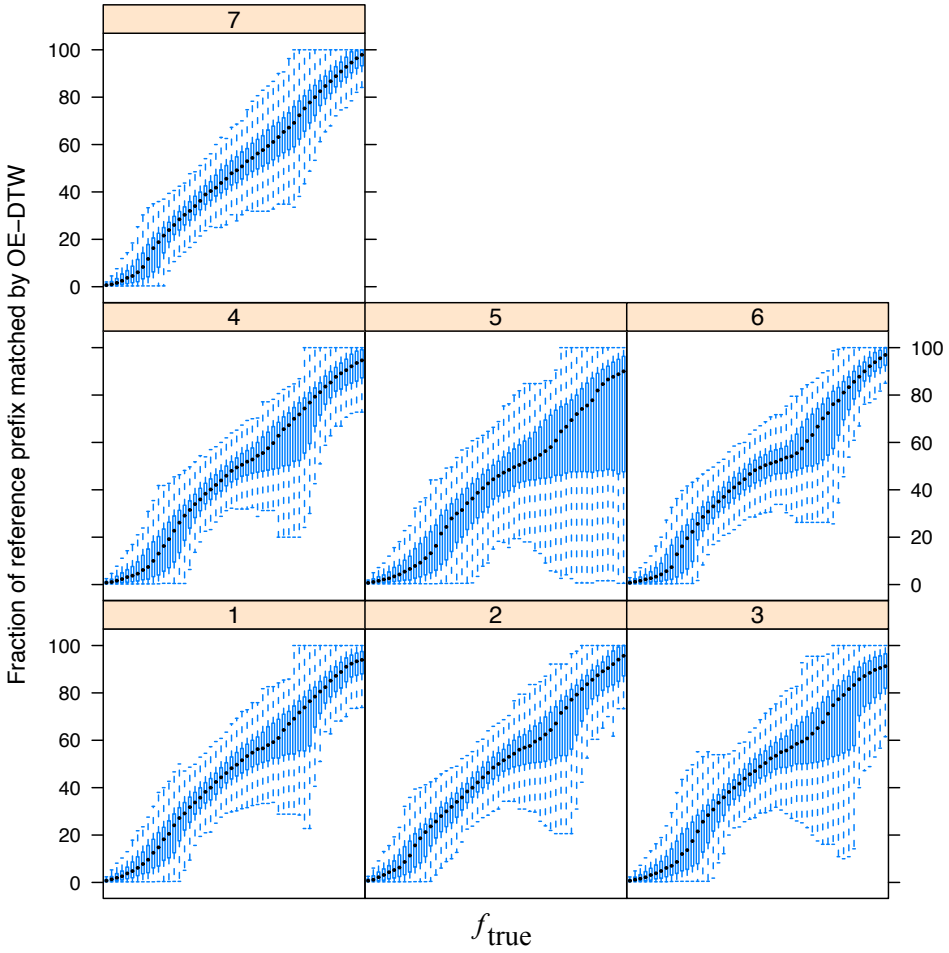

Figure 9. Heads of time series in the NR dataset, truncated at a varying fractions f_{true}, are aligned with 5 similar uncut time series. The distribution of predicted f_{OE} is plotted with respect to f_{true} (quartile bands) grouped by exercise type.

Table 9
Root mean square of $(f_{\text{OE}} - f_{\text{true}})$ over all tests performed (top row) and for median f_{OE} (bottom).

RMS	$i = 1$	2	3	4	5	6	7
All pairs	12.76	13.26	16.43	14.42	19.43	12.16	11.18
Medians	4.60	4.23	6.38	5.11	9.36	5.56	3.61

5.3.2 Match length prediction for the UCR datasets

To test the reliability of fraction recognition over a wide range of realistic problems, we repeated the experiments previously described on the 20 datasets in the UCR repository. At the time of this writing, the UCR repository contained 23,999 time series, which is too large for a comprehensive analysis with incremental alignments. We therefore selected a subset of 710 elements by randomly choosing four time series from each class of the original repository (where available).

As in Section 5.3.1 above, we formed all possible ordered pairs of homogeneous class and, for each, we computed 50 alignments truncating the test at $f_{\text{true}} = 2, 4, \ldots 100\%$. The relationship obtained between f_{true} and f_{OE} is shown in Figure 10. Each panel shows the distribution of f_{OE} for a different data set, averaged over available classes and instances, shaded areas showing quartile bands.

In general, agreement between f_{OE} and f_{true} is good for all values of f_{true} when qualitative trends in the time series are well spread out along the time axis, that is when even short queries contain enough features to provide a match. When significant patterns are shifted in time (e.g. in the case of datasets 3, 5, 10, 11, 19, 20), deviations from the linear proportionality are found on the graph where expected. Predicted f_{OE} closely agrees with f_{true}, in particular, when there is little variability among the time series within the same class, as it is the case for most of the 20 datasets.

6 Discussion

6.1 Exercise recognition in the rehabilitation application

A simple implementation of a DTW-based recognizer is possible if references for "correct exercises" and "wrong exercises" are both available, and applicable to multiple patients and sessions. In this case (call it case 1), one would label those references as separate classes, and uses OE-DTW with e.g. a k-NN algorithm to decide whether the current input is closer to a "known-good" or a "known-bad" reference. If several classes are defined for different types of incorrect executions (e.g. the 4- or 6-class problems), a feedback can be provided to the therapist concerning what error was probably done.

In practice, due to limitations of the specific strain sensor technology, generic templates valid for many patients and session can not be built. In this case (case 2), the "gold standard" would be a small set of recordings made by the patient with the supervision of the therapist for each session. Therefore, only "known good" references are available, and OE-DTW can be used to decide whether the current input stream is close to them or not. There would be no "known bad" examples, since it would be impractical and confusing to ask the patient to perform wrong movements on purpose. The system would work somewhat like an "outlier detector" (no information about the kind of error would be available).

The focus of this paper is to measure the performance of the OE-DTW algorithm alone, i.e. the reliability of the distance computation alone, isolating

it from the classification and segmentation problems. Therefore, in the experiments we assumed the "known good/bad" approach (case 1 above); each instance of the NR data set has a well-defined label (namely, index j), which is known, because it was thus recorded by a volunteer in the first place (under supervision).

Finally, one may think that, for the recognition to happen in real time, the stream has to be segmented in order to identify both the beginning and the end of the reference in the input. Interestingly, for the current rehabilitation

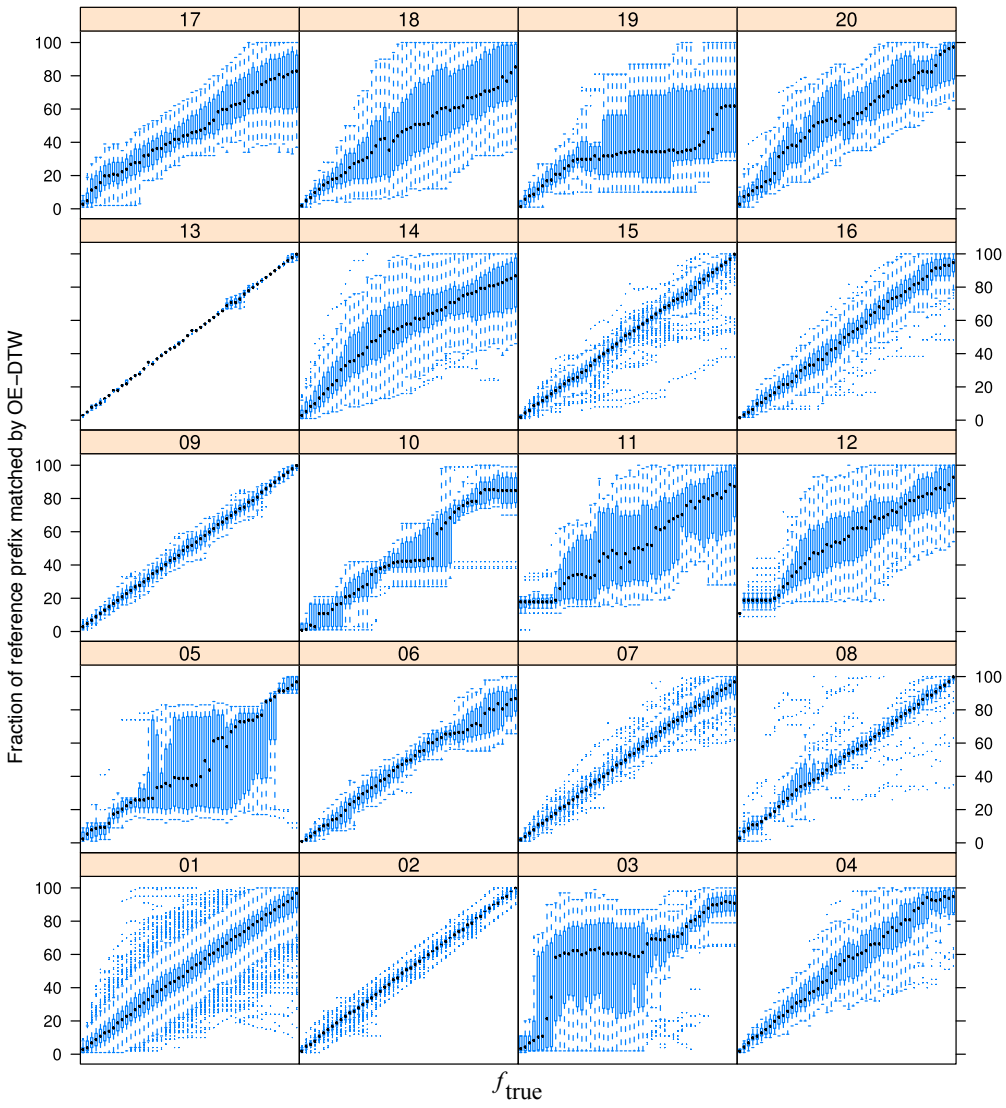

Figure 10. Prefixes of time series in the UCR datasets, truncated at a varying fraction f_{true}, are aligned with uncut time series available in the same class. OE-DTW's percentage prediction f_{OE} is plotted with respect to f_{true}, with quartile bands, grouped by data set. Deviations from linear proportionality are explained in Section 5.3.2.

application, fully unconstrained matching is not necessary. It is the system that tells the patient when to begin the exercise (after a sort of countdown), while the progress and the conclusion of the movement are recognized automatically. Nevertheless, patients can remain idle as long as they wish before starting the gesture (within a reasonable timeout); this is allowed because the initial idle segment of the input, regardless to its length, will be mapped to the initial idle segment of the reference with little or no cost. This is due to a step pattern that allows arbitrary time compressions, as explained in section 6.2. Anyway, a DTW algorithm to match segments in the middle of the input is presented and discussed in section 6.3.

6.2 Flexibility

DTW and its variants are very general techniques which have been applied to countless problems. A lot of flexibility, for example, is allowed by the choice of step patterns. Simpler step patterns (possibly with windowing imposed to limit search) are straightforward to implement, while suitably-chosen transition sets can enforce *local* constraints on the amount of compression and stretches allowed, which is a natural description for alignment constraints [16,45,46].

The asymmetric recursion (6) chosen for the recognition of rehabilitation exercises has the following useful properties:

(1) at each step, all the three productions impose the index in the test time-series to increase by one unit; therefore, the length of warping curve is always equal to the length of the input, and the warping curve is single-valued;
(2) all "rightwards" steps have a weight of 1. Therefore, normalization is easily computed dividing by the length of the input (Rabiner's slope weight type (c));
(3) the transitions have a maximum slope of 2, which is the maximum time compression allowed: the input can be at most twice as fast as the reference;
(4) the minimum slope of the warping curve is zero, which means that no limit is imposed on time dilatation. This is appropriate to allow arbitrarily long "hesitations" at the beginning and in the middle of the movement.

Although the spirit of DTW is to allow a controlled compression or expansion of time scales, some step patterns also allow deletions of "outlier" points from either time series (bottom row of Figure 3). This may happen in two cases: because of null slope weights, and when index increments greater than unity are allowed.

The asymmetric recursion used in the paper, for example, allows skipping

isolated elements in the reference, but not in the test; the asymmetry is inconsequential for smooth timeseries. On the other hand, one point of the reference may be matched by multiple consecutive elements of the test. This is possible and necessary, if one wants to allow time compressions without discarding points from the input. Step patterns can be arranged with different behaviours with this respect: R-J type III patterns, for example, allow multi-valued matches in both directions; type III forbids them [17].

Preprocessing and feature extraction, although not strictly part of the dissimilarity computation, may have a great impact on the performance of alignments. Keogh et al., for example, remark that aligning local slopes instead of raw data improves the recognition of qualitative trends [47,48]. Feature selection and extraction can be applied just as with other data mining techniques.

6.3 Both endpoints free

OE-DTW anchors test and reference timeseries at a common beginning. Depending on the problem at hand, this may not appropriate. The algorithm's constraints can be further relaxed to allow both open-begin and open-end matches (OBE-DTW); this means that one wants to discover which *inner* contiguous part of the reference is best matched by the test (Figure 11). [3]

The OBE-DTW extension is achieved appending a leftmost column to C (call it $i = 0$) and initializing it at zero:

$$C(0, j) = 0, \quad \text{for} \quad j = 1 \dots M.$$

The usual recursion, e.g. (6), is then applied to $C(i, j)$ with $i = 1 \dots N$ as above. With the initialization, one avoids the accumulation of "head mismatch" costs in $C(1, j)$. Alternatively, one can replace the initialization equation (5) with $C(1, j) = L(1, j)$ and apply (6) to elements $i \geq 2$.

To compute the optimum ending point of the alignment, again one has to normalize elements $C(N, j)$ in the rightmost column of C. For type (c) step patterns, the normalization function N is trivial, and equations (7) apply unmodified. Once j^* is found, the backtrace would start at (N, j^*) and end at $i = 1$, thus solving the problem. Therefore, patterns of type (c) are a natural choice for OBE-DTW, which can be computed with the same computational cost as DTW.

[3] The DTW naming conventions may become counter-intuitive in this case: we'll be looking for a short "needle" test in a longer "haystack" reference.

However, step patterns types (c′) and (d) can also be used, with a small increase in the number of operations. For these step patterns, ν depends both on N and on the number of elements matched in the reference, i.e. the value of $\phi_T - \phi_1$, which is unknown until the backtrace for the optimal alignment is performed. The solution is to compute M separate warping curves, labeled by $m = 1 \dots M$, after the matrix C is fully evaluated. Each warping curve is computed with a separate backtrace procedure which starts from element (N, m) and ends as soon as it reaches index $i = 1$ in the test sequence. We thus obtain a family of candidate warping curves $\Phi^{(m)} = (\psi_t^{(m)}, \phi_t^{(m)})$; now for each curve we compute the corresponding normalization factor

$$\nu_m = \nu(N, \phi_T^{(m)} - \phi_1^{(m)}).$$

The optimal alignment endpoint is finally chosen by normalizing each cumulative cost $C(N, m)$ with the corresponding constant ν_m, and minimizing:

$$\begin{aligned} m' &= \arg \min_{m=1\dots M} C(N, m)/\nu_m \\ D_{\mathrm{OBE}} &= C(N, m')/\nu_{m'} \end{aligned}$$

Performing M backtrace steps costs approximately $O(N \cdot M)$ operations, which is of the same order than the computation of C itself; if desired it can be optimized (because many paths join) or approximated.

Unlike other algorithms, OBE-DTW does not require thresholds or free parameters, and provides an exact solution. All the usual (normalizable) step patterns of DTW are available, so there are no restrictions on the relative lengths of test and reference.

6.4 Match length prediction

Series containing almost constant subsequences may cause percentage prediction to falter. This is due to the fact that a constant test matches a constant reference or any of its substrings equally well. In all the median curves of Figure 9, for example, two deviations from the diagonal are seen at $f_{\mathrm{true}} \sim 20\%$ and $f_{\mathrm{true}} \sim 70\%$. They are explained by the typical structure of the rehabilitation movement, which usually begin with a variable pause, followed by a "forward" movement, then another variable pause while holding the distal position, a backwards action which moves the arm back into the original position, and eventually another short pause in the resting position before the acquisition is over. While movement phases can be easily identified, sensors

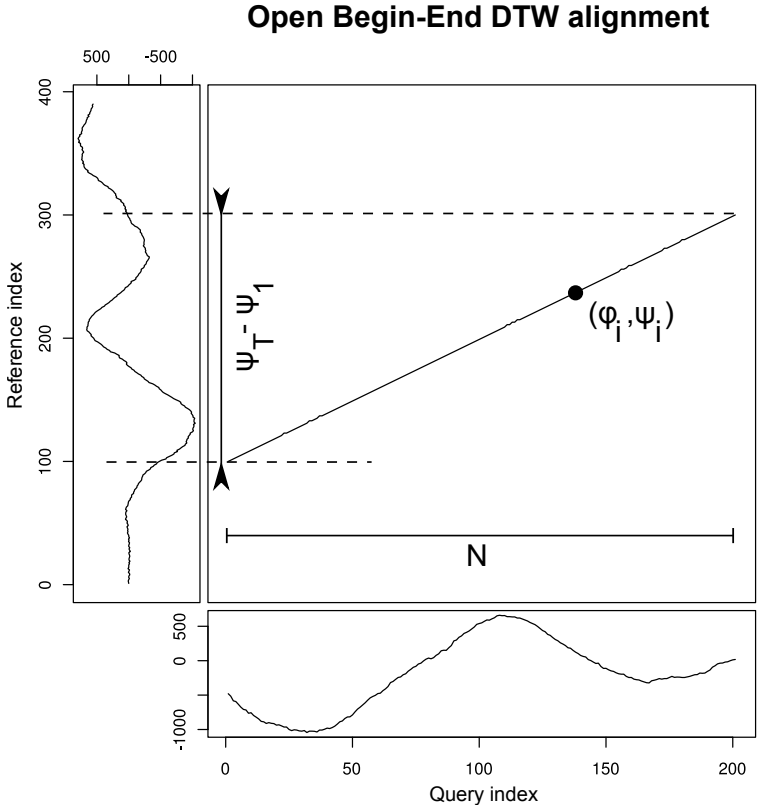

Figure 11. Warping function for OBE-DTW: both endpoints are free.

are relatively idle during the pauses and the corresponding matched fraction is not well defined.

6.5 End-point detection

When the *whole* length of a test is aligned with a reference, one expects the OE-DTW algorithm to return an almost global alignment, with the possible exception of an "idle" tail, if present in the reference (typically $\sim 5-10\%$ for the NR dataset). Statistics for f_{OE} at $f_{\mathrm{true}} = 100\%$ confirm what expected (Table 10): given complete inputs, 75% of the times f_{OE} is above 80%, and median values are above 90%. A notable exception is again exercise type $i = 5$, for which a quarter of the alignments covered 47% of the reference or less; this is consistent with the bad performance of 1-NN classifiers on this exercise. We conclude that OE-DTW could be used, in principle, as an end-point detector to trigger an "input complete" signal as soon as a growing query becomes similar to one of the patterns. This technique has analogies with continuous dynamic programming (CDP) algorithms, which typically employ global alignments on a moving time windows over the test [49,35]. Two advantages of an OE-DTW endpoint detector based on patterns with respect to an energy-based detector,

Table 10
Summary statistics for f_{OE} at endpoint $f_{\mathrm{true}} = 100\%$ in the NR dataset. Besides exercise $i = 5$, lengths of 75% of the complete timeseries are estimated over the 80%, medians over 90%, meaning that OE-DTW can reliably predict input endpoints.

i	1st Qu.	Median	Mean	3rd Qu.
1	88.0	93.9	87.0	97.5
2	87.2	95.7	86.6	100.0
3	81.9	91.3	81.5	96.5
4	87.2	94.6	85.2	99.2
5	47.7	90.0	75.4	96.3
6	92.5	96.9	92.6	100.0
7	93.2	97.9	93.2	100.0

instead, are (1) being unaffected by long low-energy segments within the input and (2) the absence of latency.

6.6 Computational complexity

A distinguishing feature of DTW is that, despite the large number of free parameters in Φ, a proper choice of step patterns allows the optimal mapping to be computed (and normalized) in $O(N \cdot M)$ time by dynamic programming (DP). A variety of techniques have been proposed to further speedup the computation: windowing [16], slope-limited step patterns [50], indexing [51], recursive decimation [52], etc.

Remarkably, OE-DTW can be computed with the same algorithm and complexity as DTW, i.e. in $N \times M$ iterations required for a single pass over the matrix C; each iteration requires a fixed number of arithmetic operations. The convenient property that dynamic programming yields solutions for several boundary conditions at once was remarked already in 1979 by Sakoe (cf. [33], Section III.A), although in a different context.

Another computationally convenient aspect of time warping concerns the incremental update of alignments as soon as additional incoming data is appended to the input being processed. Symmetrically to what was noted in Figure 4, the monotonicity constraint imposed on ψ implies that alignments for newly acquired time points can be computed recycling elements of C already computed (Figure 12), therefore with a constant count of operations per update.

As noted above, the property of the DP algorithm that allows the efficient computation of D_{OE} is almost as old as the DTW itself [33]; however, the usefulness of semi-constrained alignments seems to have received limited attention. The property was used by Rabiner et al. to *partially* relax end-point constraints (cf. [50], algorithm UE2-1) in situations when a cascaded end-point detector might have cut the time series incorrectly, and by Sakurai et al. for stream monitoring [53]. Like OBE-DTW, UE2-1 removes the constraint from both end-points. However, the focus of UE2-1 is to align complete timeseries while still allowing for a fuzzy match at any of the corners: constraints are relaxed symmetrically from *both* the test and the reference at the same time by δ time points; δ has to be small with respect to M and N to avoid large changes in the normalization constant, and degenerate warping curves like $\Phi = [\,(1, M)\,]$ (that would be allowed if e.g. $\delta = M = N$).

Mori et al. proposed a DTW variant similar to OE-DTW for a gesture recognition system from stereo-photogrammetric data [32]. Their approach focuses on one specific type (c′) step pattern whose slope is limited between 1/2 and 2 (top right of figure 3). The (c′) pattern yields a cumulative distance which is normalized along the length of the reference. However, as seen in section 6.2, a lower bound in the slope of the warping curve is an unnecessary restriction and

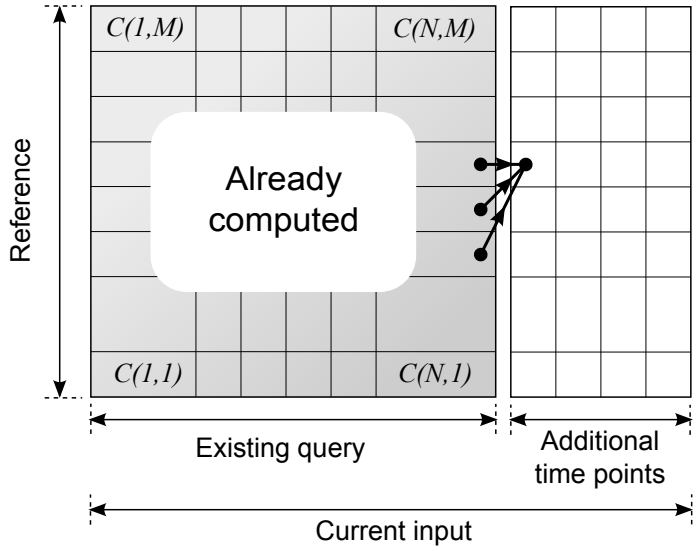

Figure 12. As soon as additional elements of the input become available, e.g. in a real-time application, an updated OE-DTW distance and alignments can be computed incrementally, therefore with a constant quota of operations per update.

may not be desirable, as it is the case in our application. A non-slope-limited step pattern of type (c′) would be problematic, because both the cumulative cost and the normalization constant of "lean" warping curves would become small, with few elements contributing to the computation. As seen in section 4.3, the OE-DTW and OBE-DTW algorithms are suitable for generic normalizable step patterns, including types (c) and (d) which are unaffected by the problem.

Hidden Markov Models (HMM) are a powerful technique for sequence analysis, useful when enough training material or background information is available to build a set of states with specific emission and transition probabilities [54]. Several authors discussed how DTW can be viewed as a HMM decoding problem [55,56].

OE-DTW and DTW have a close relationship with algorithms for string alignment, used e.g. in bioinformatics and text matching, which are also based on DP on a lattice. The Needleman-Wunsch algorithm, for example, provides *global* alignments like DTW does; Smith-Waterman discovers *local* matches, i.e. they do not constrain the warping path to pass by the endpoints of either sequence [57,58]; LCS (longest common subsequence), as implied by the name, aligns strings globally but allows outliers to be skipped. The algorithms differ by the scoring system used and their endpoints; for example, the substitution matrix, which weights the penalties for specific mismatches in aligned sites [59] plays a role analogous to the arbitrary local distance which can be defined between time series elements. Sequence alignment algorithms are usually formulated as *maximization* problems – positive elements in matrix C meaning better matches rather than, as it is for DTW, greater distance; on the contrary, mismatched elements contribute a negative quota. With this property, a lower bound is fixed on how bad an alignment can be, zero meaning "no match at all". Given the continuous nature of time series alignment problems, local distances are taken positive, so no equivalent "null point" exists for bad alignments, unless one is artificially chosen. LCS for time series alignment, for example, requires an arbitrary distance threshold [60], which is undesirable; OE-DTW, to the contrary, does not require any such threshold.

One interesting parallel between string and time series alignment lies in the penalties for deletions and insertions. Adding or removing an element in the string match corresponds to choosing a steeper or leaner warping path at a given point. String alignment algorithms (e.g. Levenshtein's edit distance) foresee arbitrary *additive* penalties for deletions and insertions, while DTW's step pattern provides *multiplicative* weights for the local distance depending on the slope, which may be null.

Finally, Latecki et al. proposed the Minimal Variance Matching (MVM) algorithm to align a given test to a reference, allowing arbitrary portions of

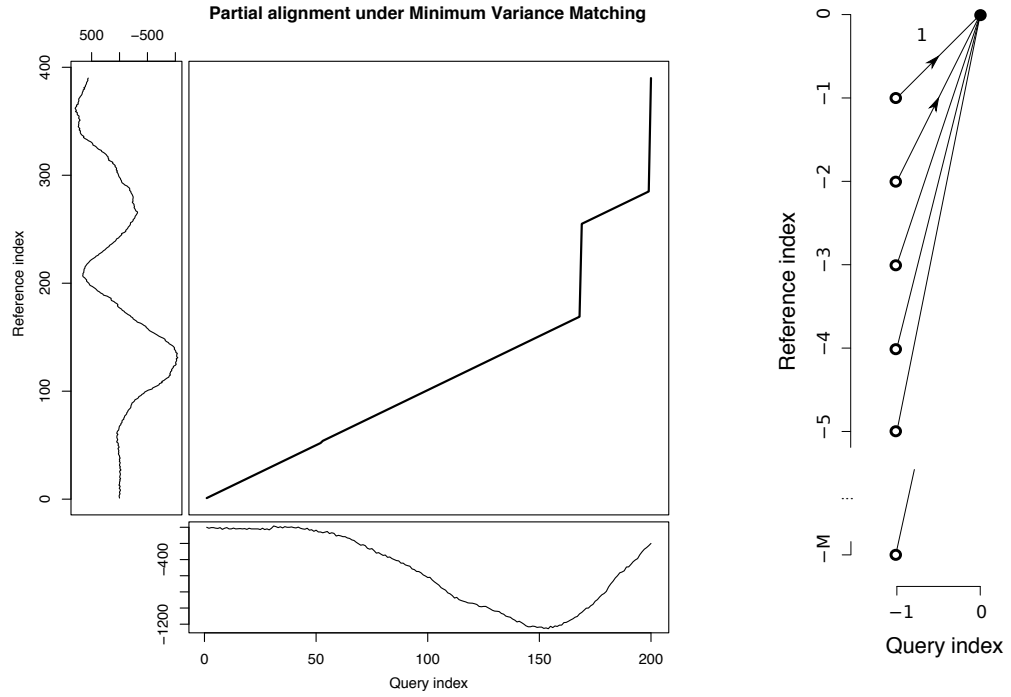

Figure 13. Left: MVM alignments on a truncated query: (1) since time compression is not allowed, the valley at query index 150 doesn't match index 125 in reference; (2) "jumps" may occur in the middle of the query, e.g. at time 175. Right: a type (c) DTW step pattern equivalent to the MVM search algorithm.

the template to be skipped [60]. Like the asymmetric step pattern used in this paper, the MVM algorithm constrains each query element to match exactly one time point in the template, but not vice-versa (so the degenerate empty-to-empty match is not allowed). Matched reference points must be in strict increasing order, but arbitrarily long discontinuities are allowed. The MVM algorithm is however not a natural solution for matching of prefixes. First, due to the strictly-increasing requirement of the reference index, the minimum local slope of the warping curve is unity, and time compression is therefore disallowed. Second, and more important, is that arbitrary "jumps" may occur in the middle of the query, depending on which part of the reference is matched best. There is therefore no direct interpretation concerning the fraction of reference matched. Finally, when the input is longer than the reference, MVM matching is undefined. Figure 13 (left) illustrates the above points on the elements NR_{1111} and NR_{1112} (fourth component) of the motor-rehabilitation dataset [13]. Interestingly, MVM may be considered a special case of DTW with a very large step pattern (right hand side of Figure 13).

This work focused on the OE-DTW distance computation algorithm alone; to evaluate its performance, we embedded it into a naive 1-NN classifier. Of course, in a real setting the algorithm should be used with more complex strategies for classification or clustering [61,55]. As already discussed, OE-DTW can be used as a basis for a real-time feedback system. The performance of a complete rehabilitation device, however, is affected by the properties of other components in the recognition pipeline, e.g. the endpoint detection, the amount and type of training data, and the classification strategies used. Of particular relevance, as mentioned in section 6.2, is the choice of an effective set of features, which is to be addressed by a-priori knowledge or through systematic experiments.

Dixon [62] proposed a forward-search approach to DTW for tracking music in real-time; the approach is analogous to Rabiner's UELM algorithm in [50], with additional constraints imposed on the slope of local path. The advantage of a limited search window is that *both* test and reference can grow arbitrarily long at a constant cost per update, whereas OE-DTW is more limited with this respect: only the input stream is allowed to grow indefinitely, as discussed in section 6.6. On the other hand, due to the local-search nature, UELM alignments can be affected by "false paths", i.e. warping curves which are optimal within the limited search window, but lead to suboptimal solutions later, when more input data becomes available. Such situations are likely in case of exercise recognition because arbitrarily long pauses, when signals are almost idle, are common.

OE-DTW is clearly asymmetric with respect to its arguments. A straightforward extension may allow "symmetrically-open" alignments, discovering the best alignment among prefixes of timeseries X with full Y, or prefixes of Y versus X. When used with a symmetrical step pattern, this would define a truly symmetrical time-warping distance between prefix time series.

An interesting issue is the matching of possibly *multiple* instances of a template in a given input query. While this is in principle a different problem than a dissimilarity computation, CDP techniques can be used to sweep over a long input with a slope-limited pattern to match one or more known references repeatedly [49,63,64,35].

Finally, abstract reasoning on time patterns in medicine requires more sophisticated abstractions over the raw timeseries data than quantitative matching [65,66]; for example, knowledge-based temporal patterns can be defined [67,68]. However, warping distances could, in principle, be embedded in such an abstraction, e.g. as a model for local matches alternative or complementary

to regression templates.

7 Conclusions

Open-end DTW generalizes the well-known dynamic time warping algorithm to the matching of truncated time series with pre-recorded templates, providing both a similarity measure and an estimate for the fraction of template matched. It can be included as a dissimilarity measure in a variety of distance-based classification and clustering strategies when early recognition is required. We have discussed how most step patterns commonly used with DTW can be used to customize the alignment, depending on the problem.

The algorithm is readily applicable, for example, in real-time applications when time series are to be matched with templates even before their recording is complete. It is currently used with success in a real-time classifier in a motor rehabilitation support device (Product Concept NR, developed in the MyHeart project [6]).

Alignments and plots in this paper have been produced by an open-source implementation of DTW for the R statistical software [42], written by the authors of this paper. The *dtw* package [69], freely available in the Comprehensive R Archive Network, computes global and open-end distances, normalizations and warping curves for most DTW variations found in the literature.

8 Acknowledgements

This work has been partially funded by the European Commission with grant EU MyHeart IST 2002-507816. The authors would like to acknowledge the support of Prof. D. De Rossi and F. Lorussi (Centro Piaggio, University of Pisa); Dr. C. Pistarini and G. Maggioni (Fondazione S. Maugeri, Pavia); and all the partners of the MyHeart project.

Author contribution statement: TG conceived the idea for the study. TG and PT contributed equally to the design and planning of the research and to the writing. PT performed experiments and analysed the data. SQ led the development of the Neuro Rehabilitation Product Concept. SQ and MS coordinated funding for the project. All authors edited and approved the final version of the manuscript.

References

[1] S. Paolucci, G. Antonucci, M. G. Grasso, D. Morelli, E. Troisi, P. Coiro, et al., Early versus delayed inpatient stroke rehabilitation: a matched comparison conducted in Italy, Archives of physical medicine and rehabilitation 81 (6) (2000) 695–700.

[2] G. Kwakkel, R. van Peppen, R. C. Wagenaar, S. W. Dauphinee, C. Richards, A. Ashburn, et al., Effects of augmented exercise therapy time after stroke: a meta-analysis, Stroke; a journal of cerebral circulation 35 (11) (2004) 2529–2539.

[3] A. Forster, J. Young, The clinical and cost effectiveness of physiotherapy in the management of elderly people following a stroke, vol. EB02, The Chartered Society Of Physiotherapy, London, UK, 2002.

[4] S. Quaglini, A. Cavallini, S. Gerzeli, G. Micieli, Economic benefit from clinical practice guideline compliance in stroke patient management, Health policy 69 (3) (2004) 305–315.

[5] D. Inzitari, The Italian Guidelines for stroke prevention. The Stroke Prevention and Educational Awareness Diffusion (SPREAD) Collaboration, Neurological sciences: official journal of the Italian Neurological Society and of the Italian Society of Clinical Neurophysiology 21 (1) (2000) 5–12.

[6] T. Giorgino, P. Tormene, B. Cattani, C. Pistarini, S. Quaglini, Remote support for stroke rehabilitation: MyHeart's Neurological Rehabilitation concept, Medinfo 2007: Proceedings of the 12th World Congress on Health (Medical) Informatics 12 (Pt 1) (2007) 92–96, K. A. Kuhn, J. R. Warren, T.-Y. Leong (eds). IOS Press, Amsterdam.

[7] T. Giorgino, P. Tormene, S. Quaglini, A Multivariate Time-Warping Based Classifier for Gesture Recognition with Wearable Strain Sensors, in: A. Dittmar, J. Clark, E. McAdams, N. Lovell (eds.), Proc. 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society EMBS 2007, IEEE Computer Society, Los Alamitos, CA, USA, 2007, pp. 4903–4906.

[8] F. Carpi, D. De Rossi, Electroactive polymer-based devices for e-textiles in biomedicine, Information Technology in Biomedicine, IEEE Transactions on 9 (3) (2005) 295–318.

[9] D. De Rossi, A. Lymberis, New generation of smart wearable health systems and applications., Information Technology in Biomedicine, IEEE Transactions on 9 (3) (2005) 293–294.

[10] T. Giorgino, F. Lorussi, D. De Rossi, S. Quaglini, Posture classification via wearable strain sensors for neurological rehabilitation, in: A. Dhawan, M. Akay, A. F. Laine, K. Chon (eds.), Engineering in Medicine and Biology Society, 2006. EMBS '06. 28th Annual International Conference of the IEEE, IEEE Computer Society, Los Alamitos, CA, USA, 2006, pp. 6273–6276.

[11] C. Mattmann, O. Amft, H. Harms, G. Troster, F. Clemens, Recognizing upper body postures using textile strain sensors, in: T. Starner, B. Schiele, T. Ryhänen (eds.), Proc. 11th IEEE International Symposium on Wearable Computers, IEEE Computer Society, Los Alamitos, CA, USA, 2007, pp. 29–36.

[12] F. Lorussi, E. P. Scilingo, M. Tesconi, A. Tognetti, D. De Rossi, Strain sensing fabric for hand posture and gesture monitoring, IEEE Transactions on Information Technology in Biomedicine 9 (3) (2005) 372–381.

[13] P. Tormene, T. Giorgino, Upper-limb rehabilitation exercises acquired through 29 elastomer strain sensors placed on fabric. release 1.0, Accessed: 30 October 2008, `http://www.labmedinfo.org/research/myheart/myheart.htm` (July 2008).

[14] F. Duchêne, C. Garbay, V. Rialle, Learning recurrent behaviors from heterogeneous multivariate time-series, Artificial intelligence in medicine 39 (1) (2007) 25–47.

[15] C. Antunes, A. Oliveira, Temporal data mining: an overview, in: K. P. Unnikrishnan, R. Uthurusamy (eds.), Proceedings of the workshop on temporal data mining at the 7th international conference on knowledge discovery and data mining (KDD'01), Association for Computing Machinery, San Francisco, CA, 2001, pp. 1–13.

[16] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing], IEEE Transactions on 26 (1) (1978) 43–49.

[17] L. Rabiner, B.-H. Juang, Fundamentals of speech recognition, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[18] L. Kaufman, P. Rousseeuw, Finding Groups in Data: an introduction to cluster analysis, Wiley-Interscience, New York, 1990.

[19] M. Faundez-Zanuy, On-line signature recognition based on VQ-DTW, Pattern Recognition 40 (3) (2007) 981–992.

[20] T. Rath, R. Manmatha, Word image matching using dynamic time warping, in: R. Manmatha (ed.), Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, IEEE Computer Society, Los Alamitos, CA, USA, 2003, pp. II–521–II–527.

[21] K. Gollmer, C. Posten, Supervision of bioprocesses using a dynamic time warping algorithm, Control Engineering Practice 4 (9) (1996) 1287–1295.

[22] B. Huang, W. Kinsner, ECG frame classification using dynamic time warping, in: W. Kinsner, A. Sebak, K. Ferens (eds.), Proc. Canadian Conference on Electrical and Computer Engineering IEEE CCECE 2002, vol. 2, IEEE Computer Society, Los Alamitos, CA, USA, 2002, pp. 1105–1110.

[23] T. Syeda-Mahmood, D. Beymer, F. Wang, Shape-based matching of ECG recordings, in: A. Dittmar, J. Clark, E. McAdams, N. Lovell (eds.), Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE, IEEE Computer Society, Los Alamitos, CA, USA, 2007, pp. 2012–2018.

[24] V. Tuzcu, S. Nas, Dynamic time warping as a novel tool in pattern recognition of ECG changes in heart rhythm disturbances, in: M. Jamshidi, M. Johnson, P. Chen (eds.), Proc. IEEE International Conference on Systems, Man and Cybernetics, vol. 1, IEEE Computer Society, Los Alamitos, CA, USA, 2005, pp. 182–186 Vol. 1.

[25] S. Casarotto, A. Bianchi, S. Cerutti, N. Vanello, E. Ricciardi, C. Gentili, et al., Combination of event-related potentials and functional magnetic resonance imaging during single-letter reading, in: A. Dhawan, M. Akay, A. F. Laine, K. Chon (eds.), Proc. 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society EMBS '06, IEEE Computer Society, Los Alamitos, CA, USA, 2006, pp. 984–987.

[26] C. D. Brina, R. Niels, A. Overvelde, G. Levi, W. Hulstijn, Dynamic time warping: a new method in the study of poor handwriting, Human movement science 27 (2) (2008) 242–255.

[27] J. Aach, G. M. Church, Aligning gene expression time series with time warping algorithms, Bioinformatics 17 (6) (2001) 495–508.

[28] F. Hermans, E. Tsiporkova, Merging microarray cell synchronization experiments through curve alignment, Bioinformatics 23 (2) (2007) 64–70.

[29] B. Kartikeyan, A. Sarkar, Shape description by time series, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (9) (1989) 977–984.

[30] L. Wei, E. Keogh, X. Xi, SAXually explicit images: Finding unusual shapes, in: C. W. Clifton, N. Zhong, J. Liu, B. W. Wah, X. Wu (eds.), Data Mining, 2006. ICDM '06. Sixth International Conference on, IEEE Computer Society, Los Alamitos, CA, USA, 2006, pp. 711–720.

[31] Y.-S. Tak, A leaf image retrieval scheme based on partial dynamic time warping and two-level filtering, in: D. Wei, T. Miyazaki, I. Paik (eds.), Proc. 7th IEEE International Conference on Computer and Information Technology CIT 2007, IEEE Computer Society, Los Alamitos, CA, USA, 2007, pp. 633–638.

[32] A. Mori, S. Uchida, R. Kurazume, R. Taniguchi, T. Hasegawa, H. Sakoe, Early recognition and prediction of gestures, in: B. Werner (ed.), Proc. 18th International Conference on Pattern Recognition ICPR 2006, vol. 3, IEEE Computer Society, Los Alamitos, CA, USA, 2006, pp. 560–563.

[33] H. Sakoe, Two-level DP-matching – A dynamic programming-based pattern matching algorithm for connected word recognition, Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing], IEEE Transactions on 27 (6) (1979) 588–595.

[34] C. Myers, L. Rabiner, A. Rosenberg, Performance tradeoffs in dynamic time warping algorithms for isolated word recognition, IEEE Trans. Acoust., Speech, Signal Process. 28 (6) (1980) 623–635.

[35] H. Wu, R. Kido, T. Shioyama, Improvement of continuous dynamic programming for human gesture recognition, in: A. Sanfeliu, J. Villanueva, M. Vanrell, R. Alquezar, A. Jain, J. Kittler (eds.), Proc. 15th International Conference on Pattern Recognition, vol. 2, IEEE Computer Society, Los Alamitos, CA, USA, 2000, pp. 945–948.

[36] E. Keogh, X. Xi, L. Wei, C. A. Ratanamahatana, The UCR Time Series Classification/Clustering Homepage, Accessed: 15 May 2007, `http://www.cs.ucr.edu/~eamonn/time_series_data/`.

[37] J. K. Martin, D. S. Hirschberg, Small sample statistics for classification error rates II: Confidence intervals and significance tests, Tech. Rep. ICS-TR-96-22, University of California, Irvine. Technical Report No. 96-22 (1996).

[38] E. P. Scilingo, F. Lorussi, A. Mazzoldi, D. De Rossi, Strain-sensing fabrics for wearable kinaesthetic-like systems, IEEE Sensors Journal 3 (4) (2003) 460–467.

[39] J. Cohen, A coefficient of agreement for nominal scales, Educational and Psychological Measurement 20 (1) (1960) 37–46.

[40] T. Kvalseth, Measurement of interobserver agreement using a "standard": measure formulation and statistical inferences, in: J. M. Tien (ed.), Proc. IEEE International Conference on Systems, Man, and Cybernetics 'Computational Cybernetics and Simulation', vol. 2, IEEE Computer Society, Los Alamitos, CA, USA, 1997, pp. 1787–1790.

[41] F. Lorussi, W. Rocchia, E. P. Scilingo, A. Tognetti, D. De Rossi, Wearable redundant fabric-based sensors arrays for reconstruction of body segment posture, IEEE Sensors Journal 4 (6) (2004) 807–818.

[42] R Development Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0 (2007).

[43] J. Demšar, Statistical Comparisons of Classifiers over Multiple Data Sets, The Journal of Machine Learning Research 7 (2006) 1–30.

[44] S. L. Salzberg, On Comparing Classifiers: Pitfalls to Avoid and a Recommended Approach, Data Mining and Knowledge Discovery 1 (3) (1997) 317–328.

[45] F. Itakura, Minimum prediction residual principle applied to speech recognition, Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing], IEEE Transactions on 23 (1) (1975) 67–72.

[46] C. S. Myers, A comparative study of several dynamic time warping algorithms for speech recognition, Master's thesis, MIT (Jun 20 1980).

[47] E. Keogh, M. Pazzani, Derivative Dynamic Time Warping, in: V. Kumar, R. Grossman (eds.), First SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2001.

[48] R. Muscillo, S. Conforto, M. Schmid, P. Caselli, T. D'Alessio, Classification of motor activities through derivative dynamic time warping applied on accelerometer data, in: A. Dittmar, J. Clark, E. McAdams, N. Lovell (eds.), Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE, IEEE Computer Society, Los Alamitos, CA, USA, 2007, pp. 4930–4933.

[49] R. Oka, Spotting Method for Classification of Real World Data, The Computer Journal 41 (8) (1998) 559–565.

[50] L. Rabiner, A. Rosenberg, S. Levinson, Considerations in dynamic time warping algorithms for discrete word recognition, IEEE Trans. Acoust., Speech, Signal Process. 26 (6) (1978) 575–582.

[51] E. Keogh, C. A. Ratanamahatana, Exact indexing of dynamic time warping, Knowledge and Information Systems 7 (3) (2005) 358–386.

[52] S. Salvador, P. Chan, FastDTW: Toward accurate dynamic time warping in linear time and space., in: K. P. Unnikrishnan, R. Uthurusamy, J. Han (eds.), KDD Workshop on Mining Temporal and Sequential Data, ACM, New York, NY, USA, 2004, pp. 70–80.

[53] Y. Sakurai, C. Faloutsos, M. Yamamuro, Stream monitoring under the time warping distance, in: L. Liu, A. Yazici, R. Chirkova, V. Oria (eds.), Proc. IEEE 23rd International Conference on Data Engineering ICDE 2007, IEEE Computer Society, Los Alamitos, CA, USA, 2007, pp. 1046–1055.

[54] L. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE 77 (2) (1989) 257–286.

[55] T. Oates, L. Firoiu, P. R. Cohen, Using dynamic time warping to bootstrap hmm-based clustering of time series, in: Sequence Learning - Paradigms, Algorithms, and Applications, Springer-Verlag, London, UK, 2001, pp. 35–52.

[56] R. Yaniv, D. Burshtein, An enhanced dynamic time warping model for improved estimation of dtw parameters, IEEE Trans. Speech Audio Process. 11 (3) (2003) 216–228.

[57] G. Navarro, A guided tour to approximate string matching, ACM Computing Surveys 33 (1) (2001) 31–88.

[58] P. Reiners, Dynamic programming and sequence alignment, Accessed: 30 October 2008, `http://www.ibm.com/developerworks/java/library/j-seqalign/index.html` (Mar 2008).

[59] S. Henikoff, J. G. Henikoff, Amino acid substitution matrices from protein blocks, Proceedings of the National Academy of Sciences of the United States of America 89 (22) (1992) 10915–10919.

[60] L. J. Latecki, V. Megalooikonomou, Q. Wang, D. Yu, An elastic partial shape matching technique, Pattern Recognition 40 (11) (2007) 3069–3080.

[61] T. Warren Liao, Clustering of time series data–a survey, Pattern Recognition 38 (11) (2005) 1857–1874.

[62] S. Dixon, An on-line time warping algorithm for tracking musical performances, in: L. P. Kaelbling, A. Saffiotti (eds.), Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, vol. IJCAI-05, Professional Book Center, Denver, CO, 2005, pp. 1727–1728.

[63] T. Nishimura, R. Oka, Spotting recognition of human gestures from time-varying images, in: M. E. Kavanaugh (ed.), Proc. Second International Conference on Automatic Face and Gesture Recognition, IEEE Computer Society, Los Alamitos, CA, USA, 1996, pp. 318–322.

[64] H. Kameya, S. Mori, R. Oka, A segmentation-free biometric writer verification method based on continuous dynamic programming, Pattern Recognition Letters 27 (6) (2006) 567–577.

[65] M. Stacey, C. McGregor, Temporal abstraction in intelligent clinical data analysis: a survey., Artificial intelligence in medicine 39 (1) (2007) 1–24.

[66] J. C. Augusto, Temporal reasoning for decision support in medicine., Artificial intelligence in medicine 33 (1) (2005) 1–24.

[67] I. J. Haimowitz, Knowledge-based trend detection and diagnosis, Ph.D. thesis, Massachusetts Institute of Technology. Dept. of Electrical Engineering and Computer Science, Cambridge, MA, USA (1994).

[68] I. J. Haimowitz, I. S. Kohane, Managing temporal worlds for medical trend diagnosis, Artificial intelligence in medicine 8 (3) (1996) 299–321.

[69] T. Giorgino, P. Tormene, dtw: Dynamic Time Warping algorithms, Accessed: 30 October 2008, `http://cran.r-project.org/web/packages/dtw/index.html` (July 2008).