The majority of machine learning algorithms assumes that objects are represented as vectors. But often the objects we want to learn on are more naturally represented by other data structures such as sequences and time series. For these representations many standard learning algorithms are unavailable. We generalize gradient-based learning algorithms to time series under dynamic time warping. To this end, we introduce elastic functions, which extend functions on time series to matrix spaces. Necessary conditions are presented under which generalized gradient learning on time series is consistent. We indicate how results carry over to arbitrary elastic distance functions and to sequences consisting of symbolic elements. Specifically, four linear classifiers are extended to time series under dynamic time warping and applied to benchmark datasets. Results indicate that generalized gradient learning via elastic functions have the potential to complement the state-of-the-art in statistical pattern recognition on time series.

1. Introduction

Statistical pattern recognition on time series finds many applications in diverse domains such as speech recognition, medical signal analysis, and recognition of gestures [4, 5]. A challenge in learning on time series consists in filtering out the effects of shifts and distortions in time. A common and widely applied approach to address invariance of shifts and distortions are elastic transformations such as dynamic time warping (DTW). Following this approach amounts in learning on time series spaces equipped with an elastic proximity measure.

In comparison to Euclidean spaces, mathematical concepts such as the derivative of a function and a well-defined addition under elastic transformations are unknown in
time series spaces. Therefore gradient-based algorithms cannot be directly applied to time series under elastic transformations.

The weak mathematical structure of time series spaces bears two consequences: (a) there are only few learning algorithms that directly operate on time series under elastic transformation; and (b) simple methods like the nearest neighbor classifier together with the DTW distance belong to the state-of-the-art and are reported to be exceptionally difficult to beat [1,10,26].

To advance the state-of-the-art in learning on time series, first adaptive methods have been proposed. They mainly devise or apply different measures of central tendency of a set of time series under dynamic time warping [9,17,18,15]. The individual approaches reported in the literature are k-means [7,12,13,16,25], self-organizing maps [23], and learning vector quantization [23]. These methods have been formulated in a problem-solving manner without a unifying theme. Consequently, there is no link to a mathematical theory that allows us to (1) place existing adaptive methods in a proper context, (2) derive adaptive methods on time series other than those based on a concept of mean, and (3) prove convergence of adaptive methods to solutions that satisfy necessary conditions of optimality.

Here we propose generalized gradient methods on time series spaces that combine the advantages of gradient information and elastic transformation such that the above issues (1)–(3) are resolved. The key idea behind this approach is the concept of elastic function. Elastic functions extend non-differentiable functions on time series to piecewise smooth functions on matrices. Then learning on time series amounts in minimizing piecewise smooth risk functionals using generalized gradient methods proposed by [3,14]. Specifically, we investigate elastic versions of logistic regression, (margin) perceptron learning, and linear support vector machine (SVM) for time series under dynamic time warping. We derive update rules and present different convergence results, in particular an elastic version of the perceptron convergence theorem. Though the main treatment focuses on univariate time series under DTW, we also show under which conditions the theory also holds for multivariate time series and sequences under arbitrary elastic transformations.

We tested the four elastic linear classifiers to all two-class problems of the UCR time series benchmark dataset [8]. The results show that elastic linear classifiers on time series behave similarly as linear classifiers on vectors. Furthermore, our findings indicate that generalized gradient learning on time series spaces have the potential to complement the state-of-the-art in statistical pattern recognition on time series, because the simplest elastic methods are competitive with the best available methods.

2. Background

This section compiles the necessary background material and serves to introduce the terminology and notations used throughout this paper. In Section 2.1 we define the DTW distance. Section 2.2 presents the problem of learning from examples, Section 2.3 introduces piecewise smooth functions, and Section 2.4 describes the generalized gradient method for minimizing piecewise smooth function.
2.1. Dynamic Time Warping Distance

By \([n]\) we denote the set \([1, \ldots, n]\) for some \(n \in \mathbb{N}\). A time series of length \(n\) is an ordered sequence \(x = (x_1, \ldots, x_n)\) with features \(x_i \in \mathbb{R}\) sampled at discrete points of time \(i \in [n]\).

To define the DTW distance between time series \(x\) and \(y\) of length \(n\) and \(m\), resp., we construct a grid \(G = [n] \times [m]\). A warping path in grid \(G\) is a sequence \(\phi = (t_1, \ldots, t_p)\) consisting of points \(t_k = (i_k, j_k) \in G\) such that

1. \(t_1 = (1, 1)\) and \(t_p = (n, m)\) (boundary conditions)
2. \(t_{k+1} - t_k \in \{(1, 0), (0, 1), (1, 1)\}\) (warping conditions)

for all \(1 \leq k < p\).

A warping path \(\phi\) defines an alignment between \(x\) and \(y\) by assigning elements \(x_i\) of sequence \(x\) to elements \(y_j\) of sequence \(y\) for all points \((i, j)\) \(\in \phi\). The boundary condition enforces that the first and last element of both time series are assigned to one another accordingly. The warping condition summarizes what is known as the monotonicity and continuity condition. The monotonicity condition demands that the points of a warping path are in strict ascending lexicographic order. The continuity condition defines the maximum step size between two successive points in a path.

The cost of aligning \(x = (x_1, \ldots, x_n)\) and \(y = (y_1, \ldots, y_m)\) along a warping path \(\phi\) is defined by

\[
d_\phi(x, y) = \sum_{(i, j) \in \phi} c(x_i, y_j),
\]

where \(c(x_i, y_j)\) is the local transformation cost of aligning features \(x_i\) and \(y_j\). Unless otherwise stated, we assume that the local transformation costs are given by \(c(x_i, y_j) = (x_i - y_j)^2\). Then the distance function

\[
d(x, y) = \min_\phi d_\phi(x, y),
\]

is the dynamic time warping (DTW) distance between \(x\) and \(y\), where the minimum is taken over all warping paths in \(G\).

2.2. The Problem of Learning

We consider learning from examples as the problem of minimizing a risk functional. To present the main ideas, it is sufficient to focus on supervised learning.

Consider an input space \(\mathcal{X}\) and output space \(\mathcal{Y}\). The problem of supervised learning is to estimate an unknown function \(f^* : \mathcal{X} \rightarrow \mathcal{Y}\) given a training set

\[
\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\} \subseteq \mathcal{X} \times \mathcal{Y},
\]

where the examples \((x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\) are drawn independent and identically distributed according to an underlying joint probability distribution \(P(x, y)\) on \(\mathcal{X} \times \mathcal{Y}\).
To measure how well a function $f : \mathcal{X} \to \mathcal{Y}$ predicts output values $y$ from $x$, we introduce the risk

$$R[f] = \int_{\mathcal{X} \times \mathcal{Y}} \ell(y, f(x)) dP(x, y),$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is a loss function that quantifies the cost of predicting $f(x)$ when the true output value is $y$.

The goal of learning is to find a function $f : \mathcal{X} \to \mathcal{Y}$ that minimizes the risk. The problem is that we cannot directly compute the risk of $f$, because the probability distribution $P(x, y)$ is unknown. But we can use the training examples to estimate the risk of $f$ by the empirical risk

$$R_N[f] = \frac{1}{N} \sum_{i=1}^N \ell(y_i, f(x_i)).$$

The empirical risk minimization principle suggests to approximate the unknown function $f_*$ by a function $f_N = \arg\min_{f \in \mathcal{F}} R_N[f]$ that minimizes the empirical risk over a fixed hypothesis space $\mathcal{F} \subset \mathcal{Y}^{\mathcal{X}}$ of functions $f : \mathcal{X} \to \mathcal{Y}$.

Under appropriate conditions on $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{F}$, the empirical risk minimization principle is justified in the following sense: (1) a minimizer $f_N$ of the empirical risk exists, though it may not be unique; and (2) the risk $R[f_N]$ converges in probability to the risk $R[f_*]$ of the best but unknown function $f_*$ when the number $N$ of training examples goes to infinity.

### 2.3. Piecewise Smooth Functions

A function $f : \mathcal{X} \to \mathbb{R}$ defined on a Euclidean space $\mathcal{X}$ is piecewise smooth, if $f$ is continuous and there is a finite collection of continuously differentiable functions

$$\mathcal{R}(f) = \{ f_i : \mathbb{R} \to \mathbb{R} : i \in \mathcal{I} \}$$

indexed by the set $\mathcal{I}$ such that

$$f(x) \in \{ f_i(x) : i \in \mathcal{I} \}$$

for all $x \in \mathcal{X}$. We call the collection $\mathcal{R}(f)$ a representation for $f$. A function $f_i \in \mathcal{R}(f)$ satisfying $f_i(x) = f(x)$ is an active function of $f$ at $x$. By $\mathcal{A}_f(x) = \{ i \in \mathcal{I} : f_i(x) = f(x) \}$ we denote the active index set of $f$ at $x$.

Piecewise smooth functions are closed under composition, scalar multiplication, finite sums, pointwise maximum and minimum operations. In particular, the maximum and minimum operations of a finite collection of differentiable functions allow us to construct piecewise smooth functions. Piecewise functions $f$ are non-differentiable
on a set of Lebesgue measure zero, that is $f$ is differentiable almost everywhere. In addition, they admit the notion of generalized gradient

$$\partial f(x) = \text{con} \{ \nabla f_i(x) : i \in A_f(x) \}$$

where con $S$ denotes the convex hull of subset $S \subseteq X$ and $\nabla f_i(x)$ is the gradient of active function $f_i$ at $x$. At differentiable points $x$, the generalized gradient is of the form $\partial f(x) = \{ \nabla f(x) \}$.

2.4. Generalized Gradient Methods

Let $Z \subseteq X$ some (sufficiently large) bounded convex constraint set. Consider the minimization problem

$$\min_{x \in Z} f(x), \quad (1)$$

where $f$ is piecewise smooth. Let $Z_*$ denote the subset of solutions satisfying the necessary condition of optimality and $f_* = \{ f(x) : x \in Z_* \}$ is the set of solution values. Consider the following iterative method, called generalized gradient method henceforth:

$$x^0 \in Z \quad (2)$$
$$x^{t+1} \in \Pi_Z (x^t - \eta^t \cdot \nabla f_i (x^t)) \quad (3)$$
$$i \in A_f (x^t), \quad (4)$$

where $\Pi_Z$ is the multi-valued projection onto $Z$ and $\eta^t$ is the learning rate satisfying the conditions

$$\lim_{t \to \infty} \eta^t = 0 \quad (5)$$
$$\sum_{t=0}^{\infty} \eta^t = \infty \quad (6)$$

The generalized gradient method minimizes a piecewise smooth function $f$ by first selecting an active index $i \in A_f(x)$ of $f$ at the current iterate $x^t$ and then performing a gradient-like update step. This procedure converges to a solution satisfying the necessary condition of optimality [3, 14]:

**Theorem 1 (Ermoliev-Norkin)** The sequence $(x^t)$ generated by method (2)–(6) converges to the solution of problem (1) in the following sense:

1. the limits points $\bar{x}$ of $(x^t)$ with minimum value $f(\bar{x})$ are contained in $Z_*$.

2. the limits points $\bar{f}$ of $(f(x^t))$ are contained in $f_*$.  

Ermoliev and Norkin also presented a consistency theorem under similar conditions when the problem is posed as that of a stochastic optimization problem [3, 14].
3. Generalized Gradient Learning on Time Series Spaces

In this section, we generalize gradient-based learning to time series spaces. Section 3.1 introduces elastic functions as a key concept. In Section 3.2, we describe a generic scheme of supervised generalized gradient learning. Section 3.3 discusses model selection. In Section 3.4, we investigate elastic linear classifiers. Then we advance to unsupervised generalized gradient learning in Section 3.5 and finally conclude with generalizations to other elastic proximity functions and arbitrary sequence data in Section 3.6.

3.1. Elastic Functions

Functions \( f : \mathcal{T} \to \mathbb{R} \) on time series under dynamic time warping are non-smooth, because the derivative of a function is undefined outside of normed vector spaces. To make derivatives available for functions on time series, we introduce the concept of elastic function, which links functions on time series to matrix spaces \( \mathcal{X} = \mathbb{R}^{n \times m} \).

The matrix space \( \mathcal{X} \) is the Euclidean space of all real \((n \times m)\)-matrices with inner product

\[
\langle X, Y \rangle = \sum_{i,j} x_{ij} \cdot y_{ij},
\]

for all \( X, Y \in \mathcal{X} \). The inner product induces the Euclidean norm

\[
\|X\| = \sqrt{\langle X, X \rangle}
\]

also known as the Frobenius norm.\(^1\) The dimension \( n \times m \) of \( \mathcal{X} \) has the following meaning: the number \( n \) of rows refers to the maximum length of all time series from the training set \( \mathcal{D} \). The number \( m \) of columns is a problem dependent complexity parameter.

We embed time series into a matrix from \( \mathcal{X} \) along a warping path as illustrated in Figure 1. Suppose that \( x = (x_1, \ldots, x_k) \) is a time series of length \( k \leq n \). By \( \mathcal{P}(x) \) we denote the set of all warping paths in the grid \( \mathcal{G} = [k] \times [m] \) defined by the length \( k \) of time series \( x \) and the number \( m \) of columns of \( \mathcal{X} \). An elastic embedding of time series \( x \) into matrix \( W = (w_{ij}) \) along warping path \( \phi \in \mathcal{P}(x) \) is a matrix \( x \odot_\phi W = (x_{ij}) \) with elements

\[
x_{ij} = \begin{cases} x_i & : (i,j) \in \phi \\ w_{ij} & : \text{otherwise} \end{cases}.
\]

In the following, we consider examples of elastic functions based on embeddings. The first two examples are fundamental for extending a broad class of gradient-learning algorithms to time series spaces.

**Example 1 (Elastic Euclidean Distance)** The elastic Euclidean distance is a distance function between a time series and a matrix defined by

\[
\delta : \mathcal{T} \times \mathcal{X} \to \mathbb{R}, \quad (x, Y) \mapsto \min_{\phi \in \mathcal{P}} \|x \odot_\phi Y - Y\|,
\]

\(^{1}\)We call \( \|X\| \) Euclidean norm to emphasize that we regard \( \mathcal{X} \) as a Euclidean space.
Figure 1: Embedding of time series $\mathbf{x} = (2, 4, 3, 1)$ into matrix $\mathbf{W}$ along warping path $\phi$. Left: warping path $\phi$ highlighted in the grid $\mathcal{G}$. Middle: original matrix $\mathbf{W}$. Right: matrix after embedding $\mathbf{x}$ into $\mathbf{W}$ along $\phi$. We assume that the length of the longest time series in the training set is $n = 7$. Therefore the matrix $\mathbf{W}$ has $n = 7$ rows. The number $m$ of columns of $\mathbf{W}$ is a problem dependent parameter and set to $m = 5$ in this example. Since time series $\mathbf{x}$ has length $k = 4$, the grid $\mathcal{G} = [k] \times [m]$ containing all feasible warping paths consists of $k$ rows and $m$ columns. Grids $\mathcal{G}$ vary only in the number $k$ of rows in accordance with the length $k \leq n$ of the time series to be embedded, but always have $m = 5$ columns, because matrix $\mathbf{W}$ has $m = 5$ columns.

where $\mathbf{x} \otimes_\phi \mathbf{Y}$ is the matrix obtained by embedding $\mathbf{x}$ into $\mathbf{Y}$ along $\phi$. See Figure 2 for an illustration.

Example 2 (Elastic Inner Product) The elastic inner product is a similarity function between a time series and a matrix defined by

$$\sigma : \mathcal{T} \times \mathcal{X} \to \mathcal{X}, \quad (\mathbf{x}, \mathbf{W}) \mapsto \max_{\phi \in \mathcal{P}(\mathbf{x})} \langle \mathbf{x} \otimes_\phi \mathbf{0}, \mathbf{W} \rangle.$$

Observe that $\phi$ embeds time series $\mathbf{x}$ into the zero-matrix $\mathbf{0}$.

The elastic Euclidean distance and elastic inner product are elastic proximities closely related to the DTW distance. The elastic Euclidean distance generalizes the DTW distance, and both elastic proximities have the same time and storage complexity as the DTW distance.

The next examples extend functions on vectors to elastic functions on time series by applying the elastic proximities in Example 1 and 2.

Example 3 (Elastic Linear Function) Let $\Theta = \mathcal{X} \times \mathbb{R}$ be a set of parameters. An elastic linear function is a function of the form

$$f_\theta : \mathcal{T} \to \mathbb{R}, \quad \mathbf{x} \mapsto b + \sigma(\mathbf{x}, \mathbf{W}),$$

with parameter $\theta = (\mathbf{W}, b)$, where $\mathbf{W}$ is the weight matrix and $b$ is the bias.
Figure 2: Elastic Euclidean distance $\delta(x, W)$. From left to right: time series $x = (2, 4, 3, 1)$, matrix $W$, matrix $x \otimes_\phi W$ obtained by embedding $x$ into matrix $W$ along optimal warping path $\phi$, and distance computation by aggregating the local costs giving $\delta(x, W) = \sqrt{19}$. The optimal path is highlighted in orange in $W$ and $x \otimes_\phi W$. Gray shaded areas in both matrices refer to parts that are not used, because the length $k$ of $x$ is less than $n = 7$. Since $x$ is embedded into $W$ only elements lying on the path $\phi$ contribute to the distance. All other local cost between elements of $W$ and $x \otimes_\phi W$ are zero.

**Example 4 (Single-Layer Neural Network)** Let $\Theta = \mathcal{X}^r \times \mathbb{R}^{2r+1}$ be a set of parameters. Consider the function

$$f_\theta : \mathcal{T} \to \mathbb{R}, \quad x \mapsto b + \sum_{i=1}^r w_i \sigma(f_i(x)),$$

where $\sigma$ is a sigmoid function, $f_i = f_{\theta_i}$ are elastic linear functions with parameters $\theta_i = (W_i, b_i)$, and $\theta = (\theta_1, \ldots, \theta_r, w_1, \ldots, w_r, b)$ summarizes all parameters. The function $f_\theta$ implements a neural network for time series with $r$ sigmoid units in the hidden layer and a single linear unit in the output layer.

**Example 5** Let $\Theta = \mathcal{X}^K$ be the set of parameters. Consider the function

$$f_\theta : \mathcal{T} \to \mathbb{R}, \quad x \mapsto \min \{ \delta(x, Y_1), \ldots, \delta(x, Y_K) \}$$

with parameter $\theta = (Y_1, \ldots, Y_K)$.

With the examples in mind, we derive a definition of elastic functions. We begin with the simplest form of an elastic function as a function on time series parametrized by a single matrix. Concretely, let $\Theta = \mathcal{X}$ be a set of parameters. Suppose that $F_\theta : \mathcal{X} \to \mathbb{R}$ is a function defined on the Euclidean space $\mathcal{X}$ with parameter $\theta$. An elastic function of $F_\theta$ based on matrix $Z$ is a function $f_\theta : \mathcal{T} \to \mathbb{R}$ such that

$$f_\theta(x) \in \mathcal{R}(f_\theta, x) = \{ F_\theta(x \otimes_\phi Z) : \phi \in \mathcal{P}(x) \}$$

(7)
for all \( x \in \mathcal{T} \).

We show that parametrized versions of Example 1 and 2 are elastic functions. To see this, we consider the Euclidean distance \( D_Y(X) = \|X - Y\| \) as a function of \( X \) parametrized by \( Y \). Then the function

\[
\delta_Y(x) = \min_{\phi \in P(x)} \|x \otimes_\phi Y - Y\| = \min_{\phi \in P(x)} D_Y (x \otimes_\phi Y)
\]

satisfies \( \delta_Y(x) \in R \), where

\[
R(\delta_Y, x) = \{D_Y (x \otimes_\phi Y) : \phi \in P(x)\}.
\]

This shows that the parametrized elastic Euclidean \( \delta_Y \) distance is an elastic function of the parametrized Euclidean distance \( D_Y(X) \) based on matrix \( Y \). In this example, parameter matrix \( \theta \) and base matrix \( Z \) coincide.

Similarly, the parametrized elastic inner product \( \sigma_W \) is an elastic function of the parametrized inner product \( S_W \) based on the zero-matrix \( 0 \). This example shows that parameter and base matrix may differ.

Next, we extend the definition of an elastic function to the general case. For this we assume that \( \Theta = X^r \times R^s \) is a set of parameters, where \( r \geq 1 \) denotes the number of parameter matrices and \( s \geq 0 \) is the number of real-valued parameters. Let \( F_\theta : X \rightarrow R \) be a function of the form

\[
F_\theta(X) = G \left( F_1(X), \ldots, F_r(X) \right),
\]

where \( G : R^r \rightarrow R \) and \( F_1, \ldots, F_r : X \rightarrow R \) are functions. An elastic function of \( F_\theta \) based on matrices \( Z_1, \ldots, Z_r \) is a function \( f_\theta : T \rightarrow R \) such that \( f_\theta(x) \in R(\theta, x) \) for all \( x \in \mathcal{T} \), where

\[
R(\theta, x) = \left\{ G \left( F_1(x \otimes_{\phi_1} Z_1), \ldots, F_r(x \otimes_{\phi_r} Z_r) \right) : \phi_1, \ldots, \phi_r \in P(x) \right\}.
\]

The first definition of an elastic function with parameter set \( \Theta = X \) is a special case of the general definition of an elastic function. This can be seen by setting \( G = \text{id} \) and \( F = F_1 \). Thus, Examples 1 and 2 are elastic functions according to the general definition. By construction, Examples 3–5 are elastic functions.

### 3.2. Supervised Generalized Gradient Learning

This section introduces a generic scheme of generalized gradient learning for time series under dynamic time warping.

Let \( \Theta = X^r \times R^s \) be a set of parameters. Consider a hypothesis space \( \mathcal{F} \) of functions \( f_\theta : T \rightarrow Y \) with parameter \( \theta = (W_1, \ldots, W_r, b) \in \Theta \). Suppose that \( \mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\} \subseteq T \times Y \) is a training set. According to the empirical risk minimization principle, the goal is to minimize

\[
R_N[\theta] = R_N[f_\theta] = \sum_{i=1}^N \ell(y, f_\theta(x))
\]
as a function of the parameter \( \theta \). The loss \( \ell(y, f_\theta(x)) \) as a function of \( \theta \) with parameter \( z = (x, y) \) can be written as

\[
\ell_z : \Theta \to \mathbb{R}, \quad \theta \mapsto \ell(y, f_\theta(x)).
\]  
(8)

We assume that the loss \( \ell_z \) is piecewise smooth with representation

\[
\mathcal{R}(\ell_z) = \{ \ell_\Phi : \Theta \to \mathbb{R} : \Phi = (\phi_1, \ldots, \phi_r) \in \mathcal{P}^r(x) \}
\]

indexed by \( r \)-tuples of warping paths from \( \mathcal{P}(x) \). The gradient \( \nabla \ell_\Phi \) of an active function \( \ell_\Phi \) at \( \theta \) is given by

\[
\nabla \ell_\Phi = \left( \frac{\partial \ell_\Phi}{\partial W_1}, \ldots, \frac{\partial \ell_\Phi}{\partial W_r}, \frac{\partial \ell_\Phi}{\partial b} \right),
\]

where \( \frac{\partial \ell_\Phi}{\partial \theta} \) denotes the partial derivative of \( \ell_\Phi \) with respect to \( \theta \). The incremental update rule of the generalized gradient method is of the form

\[
W_i^{t+1} = W_i^t - \eta_t \frac{\partial \ell_\Phi}{\partial W_i}(\theta^t)
\]

(9)

\[
b^{t+1} = b^t - \eta_t \frac{\partial \ell_\Phi}{\partial b}(\theta^t)
\]

(10)

for all \( i \in [r] \). Under the conditions of the Ermoliev-Norkin Theorem, the generalized gradient method for minimizing the empirical risk with loss function (8) using the batch version of update rule (9) and (10) converges to solutions satisfying necessary conditions of optimality.

3.3. Model Selection

If the hypothesis space \( \mathcal{F} \) is too complex for the given training set \( \mathcal{D} \), the learned model \( f_N \) may have poor generalization performance (overfit). A common way to prevent overfitting consists in minimizing the regularized risk

\[
\hat{R}_N[f] = R_N[f] + \lambda \Omega(f),
\]

where \( \Omega(f) \) is the regularizer that punishes overly complex functions and \( \lambda \geq 0 \) is the regularization parameter that controls the trade-off between minimizing the empirical risk \( R_N[f] \) and the complexity of \( f \) quantified by \( \Omega(f) \).

Piecewise smooth regularizers for primitive functions \( \mathcal{F} \) on feature vectors directly carry over to piecewise smooth regularizers on elastic functions \( f \) of \( \mathcal{F} \) on time series. Since piecewise smoothness is closed under addition, the regularized risk is piecewise smooth whenever the empirical risk and the regularizer are piecewise smooth. As an example consider the \( L_2 \)-regularizer for functions \( f_\theta \in \mathcal{F} \) parametrized by a single matrix \( \theta \in \mathcal{X} \). Then

\[
\Omega(f_\theta) = ||\theta||^2
\]
Elastic Logistic Regression

\( Y = \{0, 1\} \)

logistic function

\[ g_\theta(x) = \frac{1}{1 + \exp(-f_\theta(x))} \]

loss function

\[ \ell = -y \log(g_\theta(x)) - (1 - y) \log(1 - g_\theta(x)) \]

partial derivative

\[ \partial_W \ell = -(y - g_\theta(x)) \cdot X \]

Elastic Perceptron

\( Y = \{\pm 1\} \)

loss function

\[ \ell = \max\{0, -y \cdot f_\theta(x)\} \]

partial derivative

\[ \partial_W \ell = -y \cdot X \cdot I\{\ell > 0\} \]

Elastic Margin Perceptron

\( Y = \{\pm 1\} \)

loss function

\[ \ell = \max\{0, \xi - y \cdot f_\theta(x)\} \]

partial derivative

\[ \partial_W \ell = -y \cdot X \cdot I\{\ell > 0\} \]

Elastic Linear SVM

\( Y = \{\pm 1\} \)

loss function

\[ \ell = \lambda \|W\|^2 + \max\{0, 1 - y \cdot f_\theta(x)\} \]

partial derivative

\[ \partial_W \ell = -y \cdot X \cdot I\{\ell > 0\} \]

Table 1: Examples of elastic linear classifiers. By \( \partial_W \ell \) we denote a partial derivative of an active function of \( \ell \) with respect to \( W \). The partial derivatives \( \partial_\ell \) coincide with their corresponding counterparts in vector spaces and are therefore not included. The matrix \( X = x \otimes \phi \) is obtained by embedding time series \( x \) into the zero-matrix \( 0 \) along active warping path \( \phi \). The indicator function \( I\{z\} \) returns 1 if the boolean expression \( z \) is true and returns 0, otherwise. The elastic perceptron is a special case of elastic margin perceptron with margin \( \xi = 0 \). The elastic linear SVM can be regarded as a special \( L_2 \)-regularized elastic margin perceptron with margin \( \xi = 1 \).

where \( \|\cdot\| \) denotes the Euclidean (Frobenius) norm. Note that the squared Euclidean norm is smooth and the \( L_1 \) norm is piecewise smooth.

Other options for model selection are imposing global constraints such as the Sakoe-Chiba band [21] that restricts the set of feasible warping paths to a band of certain width along the main diagonal of the grid. Such global constraints has been primarily introduced to improve computational efficiency. Recently, bandwidths have been treated as hyper-parameters and tuned for improving classification accuracy. Actually, these bandwidths can be treated as parameters that control the capacity of a classifier.

For elastic function, the number of columns of the matrix space \( X \) is a similar parameter that controls the capacity as well as the computational complexity of a classifier.
3.4. Elastic Linear Classifiers

Let $Y = \{-1, 1\}$ be the output space consisting of two class labels. An elastic linear classifier is a function of the form

$$h_\theta : \mathcal{T} \to Y, \quad x \mapsto \begin{cases} +1 & : f_\theta(x) \geq 0 \\ -1 & : f_\theta(x) < 0 \end{cases}$$

(11)

where $f_\theta(x) = b + \sigma(x, W)$ is an elastic linear function and $\theta = (W, b)$ summarizes the parameters. We assign a time series $x$ to the positive class if $f_\theta(x) \geq 0$ and to the negative class otherwise.

Depending on the choice of loss function $\ell(y, f_\theta(x))$, we obtain different elastic linear classifiers as shown in Table 1.

The loss function of elastic logistic regression is differentiable as a function of $f_\theta$ and $b$, but piecewise smooth as a function of $W$. All other loss functions are piecewise smooth as a function of $f_\theta, b$ and $W$.

From the partial derivatives, we can construct the update rule of the generalized gradient method. For example, the incremental version of the generalized gradient method for the elastic perceptron loss is of the form

$$W_{t+1} = W_t + \eta_t y X$$
$$b_{t+1} = b_t + \eta_t y,$$

(12) (13)

where $X = x \otimes_\alpha 0$ and $(x, y)$ is the training example at iteration $t$. From the factor $I_{\ell > 0}$ shown in Table 1 follows that update rules (12) and (13) are only applied when $x$ is misclassified, that is when $y f_\theta(x) < 0$.

We present three convergence results. A proof is given in Appendix A.

Convergence of generalized gradient method. The generalized gradient method for minimizing the empirical risk of an elastic linear classifier with convex loss converges to a local minimum under the assumptions of the Ermoliev-Norkin Theorem.

Convergence of incremental generalized gradient method. The incremental generalized gradient method converges to a local minimum of the empirical risk of an elastic linear classifier with convex loss under the assumptions of the Ermoliev-Norkin Theorem.

Elastic margin perceptron convergence theorem. The perceptron convergence theorem states that the perceptron algorithm with constant learning rate finds a separating hyperplane, whenever the training patterns are linearly separable. A similar result holds for the elastic margin perceptron algorithm.

A finite training set $\mathcal{D} \subseteq \mathcal{T} \times Y$ is elastic-linearly separable, if there are parameters $\theta = (W, b)$ such that $h_{\theta}(x) = y$ for all training examples $(x, y) \in \mathcal{D}$. We say, $\mathcal{D}$ is elastic-linearly separable with margin $\xi > 0$ if

$$\min_{(x,y) \in \mathcal{D}} y (b + \sigma(x, W)) \geq \xi.$$

Then the following convergence theorem holds:
Theorem 2 (Elastic Margin Perceptron Convergence Theorem) Suppose that \( D \subseteq T \times Y \) is elastic-linearly separable with margin \( \xi > 0 \). Then the elastic margin perceptron algorithm with fixed learning rate \( \eta \) and margin-parameter \( \lambda \leq \xi \) converges to a solution \((W, b)\) that correctly classifies the training examples from \( D \) after a finite number of update steps, provided the learning rate is chosen sufficiently small.

3.5. Unsupervised Generalized Gradient Learning

Several unsupervised learning algorithms such as, for example, k-means, self-organizing maps, principal component analysis, and mixture of Gaussians are based on the concept of (weighted) mean. Once we know how to average a set of time series, extension of mean-based learning methods to time series follows the same rules as for vectors. Therefore, it is sufficient to focus on the problem of averaging a set of time series.

Suppose that \( D = \{x_1, \ldots, x_N\} \subset T \) is a set of unlabeled time series. Consider the sum of squared distances

\[
 f(Y) = \sum_{i=1}^{N} \delta^2(x_i, Y),
\]

where \( Y \in X \) is a matrix, and \( \delta \) is the elastic Euclidean distance. A matrix \( Y^* \) that minimizes \( f \) is an elastic mean of the set \( D \) and the minimum value \( f_* = f(Y_*) \) is the variation of \( D \).

The update rule of the generalized gradient method is of the form

\[
 Y^{t+1} = Y^t - \eta^t \sum_{i=1}^{N} (X_i - Y^t),
\]

where \( X_i = x_i \otimes_{\phi_i} Y^t \) is the matrix obtained by embedding the \( i \)-th training example \( x_i \) into matrix \( Y^t \) along active warping path \( \phi_i \). Under the conditions of the Ernolle-Norkin Theorem, the generalized gradient method for minimizing the \( f \) using update rule (15) is consistent in the mean and variation.

We consider the special case, when the learning rate is constant and takes the form \( \eta^t = 1/N \) for all \( t \geq 0 \). Then update rule (15) is equivalent to

\[
 Y^{t+1} = \frac{1}{N} \sum_{i=1}^{N} X_i,
\]

where \( X_i \) is as in (15).

3.6. Generalizations

This section identifies some generalizations of the concept of elastic functions.
3.6.1. Generalization to other Elastic Distance Functions

Elastic functions as introduced here are based on the DTW distance via embeddings along a set of feasible warping paths with squared differences as local transformation costs. The choice of distance function and local transformation cost is arbitrary. We can equally well define elastic functions based on proximities other than the DTW distance. Results on learning carry over whenever a proximity \( \rho \) on time series satisfies the following sufficient conditions: (1) \( \rho \) minimizes the costs over a set of feasible paths, (2) the cost of a feasible path is a piecewise smooth function as a function of the local transformation costs, and (3) the local transformation costs are piecewise smooth.

With regard to the DTW distance, these generalizations include the Euclidean distance and DTW distances with additional constraints such as the Sakoe-Chiba band [21]. Furthermore, absolute differences as local transformation cost are feasible, because the absolute value function is piecewise smooth.

3.6.2. Generalization to Multivariate Time Series

A multivariate time series is an ordered sequence \( x = (x_1, \ldots, x_n) \) consisting of feature vectors \( x_i \in \mathbb{R}^d \). We can define the DTW distance between multivariate time series \( x \) and \( y \) as in the univariate case but replace the local transformation cost \( c(x_i, y_j) = (x_i - y_j)^2 \) by \( c(x_i, y_j) = \|x_i - y_j\|^2 \).

To define elastic functions, we embed multivariate time series into the set \( \mathcal{X} = (\mathbb{R}^d)^{n \times m} \) of vector-valued matrices \( X = (x_{ij}) \) with elements \( x_{ij} \in \mathbb{R}^d \). These adjustment preserve piecewise smoothness, because the Euclidean space \( \mathcal{X} \) is a direct product of lower-dimensional Euclidean spaces.

3.6.3. Generalization to Sequences with Symbolic Attributes

We consider sequences \( x = (x_1, \ldots, x_n) \) with attributes \( x_i \) from some finite set \( \mathcal{A} \) of \( d \) attributes (symbols). Since \( \mathcal{A} \) is finite, we can represent its attributes \( a \in \mathcal{A} \) by \( d \)-dimensional binary vectors \( a \in \{0, 1\}^d \), where all but one element is zero. The unique non-zero element has value one and is related to attribute \( a \). In doing so, we can reduce the case of attributed sequences to the case of multivariate time series.

We can introduce the following local transformation costs

\[
c(x_i, y_j) = \begin{cases} 0 & : x_i = y_j \\ 1 & : x_i \neq y_j \end{cases}.
\]

More generally, we can define local transformation costs of the form

\[
c(x_i, y_j) = k(x_i, x_i) - 2k(x_i, y_j) + k(y_j, y_j),
\]

where \( k : \mathcal{A} \times \mathcal{A} \rightarrow \mathbb{R} \) is a positive-definite kernel. Provided that the kernel is an inner product in some finite-dimensional feature space, we can reduce this generalization also to the case of multivariate time series.
Algorithm 1 (Mean Computation)

Input:
- sample $D = \{x_1, \ldots, x_N\} \subseteq T$

Procedure:
1. initialize $Y \in Z$
2. repeat
   2.1. determine active warping paths $\phi_i$ that embed $x_i$ into $Y$
   2.2. update $Y \leftarrow \nu(Y, x_1, \ldots, x_N, \phi_1, \ldots, \phi_N)$
   2.3. project $Y \leftarrow \pi(Y)$ to $Z$
until convergence

Return:
- approximation $y$ of mean

4. Relationship to Previous Approaches

Previous work on adaptive methods either focus on computing or are based on a concept of (weighted) mean of a set of time series. Most of the literature is summarized in [9, 15, 16, 23]. To place those approaches into the framework of elastic functions, it is sufficient to consider the problem of computing a mean of a set of time series.

Suppose that $D = \{x_1, \ldots, x_N\}$ is a set of time series. Consider the sum of squared distances $g(y) = \sum_{i=1}^{N} d^2(x_i, y)$, where $d$ is the DTW distance.

Algorithm 1 outlines a unifying minimization procedure of $g$. The set $Z$ in line 1 of the procedure consists of all matrices with $n$ identical rows, where $n$ is the maximum length of all time series from $D$. Thus, there is a one-to-one correspondence between time series from $T$ and matrices from $Z$. By construction, we have $g(y) = f(Y)$, where $Y \in Z$ is the matrix with all rows equal to $y$ and $f(Y)$ is the elastic version of $g$ defined in eq. (14).

In line 2.1, we determine active warping paths of the elastic function $f(Y)$ that embed $x_i$ into matrix $Y$. By construction this step is equivalent to computing optimal warping paths for determining the DTW distance between $x_i$ and $y$. Line 2.2 updates matrix $Y$ and line 2.3 projects the updated matrix $Y$ to the feasible set $Z$. The last step is equivalent to constructing a time series from a matrix.

Previous approaches differ in the form of update rule $\nu$ in line 2.2 and the projection $\pi$ in line 2.3. Algorithmically, steps 2.2 and 2.3 usually form a single step in the sense that the composition $\psi = \pi \circ \nu$ can not as clearly decomposed in two separate precessing steps as described in Algorithm 1. The choice of $\nu$ and $\pi$ is critical for convergence analysis. Problems arise when the map $\nu$ does not select a generalized
| Dataset                     | #(Train) | #(Test) | Length | $\rho$  |
|----------------------------|----------|---------|--------|---------|
| Coffee                     | 28       | 28      | 286    | 0.098   |
| ECG200                     | 100      | 100     | 96     | 1.042   |
| ECGFiveDays                | 23       | 861     | 136    | 0.169   |
| Gun Point                  | 50       | 150     | 150    | 0.333   |
| ItalyPowerDemand           | 67       | 1,029   | 24     | 2.792   |
| Lightning 2                | 60       | 61      | 637    | 0.094   |
| MoteStrain                 | 20       | 1,252   | 84     | 0.238   |
| SonyAIBORobotSurface       | 20       | 601     | 70     | 0.286   |
| SonyAIBORobotSurfaceII     | 27       | 953     | 65     | 0.415   |
| TwoLeadECG                 | 23       | 1,139   | 82     | 0.280   |
| Wafer                      | 1,000    | 6,174   | 152    | 6.579   |
| Yoga                       | 300      | 3,000   | 426    | 0.704   |

Table 2: Characteristic features of data sets for two-class classification problems. The last column shows the ratio $\rho = \text{length}/\#(\text{train})$.

...gradient and the projection $\pi$ does not map a matrix from $X$ to a closest matrix from $Y$. In these cases, it may be unclear how to define necessary conditions of optimality for the function $g$. As a consequence, even if steps 2.2 and 2.3 minimize $g$, we do not know whether Algorithm 1 converges to a local minimum of $g$. The same problems arise when studying the asymptotic properties of the mean as a minimizer of $g$.

The situation is different for the elastic function $f$. When minimizing $f$, the set $Z$ coincides with $X$. Since the function $f$ is piecewise smooth, the map $\nu$ in line 2.2 corresponds to an update step of the generalized gradient method. The projection $\pi$ in line 2.3 is the identity. The Ermoliev-Norkin Theorem provides sufficient conditions under which Algorithm 1 is consistent.

5. Experiments

The goal of this section is to assess the performance of all four elastic linear classifiers presented in Section 3.4.

5.1. Experimental Setup.

We considered all two-class problems of the UCR time series datasets [8]. Table 2 summarizes characteristic features of the datasets.

We compared the four elastic linear classifiers of Section 3.4 against different variants of the nearest neighbor (NN) classifier with DTW distance. The variants of the NN classifiers differ in the choice of prototypes. The first variant uses all training examples as prototypes. The second and third variant learned one prototype per class from the training set using k-means as second variant and agglomerative hierarchical clustering as third variant [16].

The settings of the elastic linear classifiers were as follows: The dimension of the matrix space $X$ was set to $n \times 0.1n$, where $n$ is the length of the shortest (= longest) time series in the training set. Bias and weight matrix were initialized by drawing random numbers from the uniform distribution on the interval $[-0.01, +0.01]$. Parameters
were selected by $k$-fold cross validation on the training set of size $N$. We set $k = 10$ if $N > 30$ and $k = N$ otherwise. The following parameters were selected: learning rate $\eta$ for all elastic linear classifiers, margin $\xi$ for elastic margin perceptron, and regularization parameter $\lambda$ for elastic linear SVM. The parameters were selected from the following values

$$\eta \in \{2^{-10}, 2^{-9}, \ldots, 2^0\}, \quad \xi \in \{10^{-7}, 10^{-6}, \ldots, 10^1\}, \quad \lambda \in \{2^{-10}, 2^{-9}, \ldots, 2^{-1}\}.$$ 

The final model was obtained by training the elastic linear classifiers on the whole training set using the optimal parameter(s). We assessed the generalization performance by applying the learned model to the test data. Since the performance of elastic linear classifiers depends on the random initialization of the bias and weight matrix, we repeated the last two steps 100 times, using the same selected parameters in each trial.

5.2. Results and Discussion.

Table 3 summarizes the error rates of all elastic linear classifiers and nearest neighbor classifiers.

Comparison of elastic linear classifiers and nearest neighbor methods is motivated by the following reasons: First, nearest neighbor classifiers belong to the state-of-the-art and are considered to be exceptionally difficult to beat [1, 10, 26]. Second, in Euclidean spaces linear classifiers and nearest neighbors are two simple but complementary approaches. Linear classifiers are computationally efficient, make strong assumptions about the data and therefore may yield stable but possibly inaccurate predictions. In contrast, nearest neighbor methods make very mild assumption about the data and therefore often yield accurate but possibly unstable predictions [6].

The first key observation suggests that overall generalization performance of elastic linear (EL) classifiers is comparable to the state-of-the-art NN classifier. This observation is supported by the same same number of green shaded rows (EL is better) and red shaded rows (NN is better) in Table 3. As reported by [10], ensemble classifiers of different elastic distance measures are assumed to be first approach that significantly outperformed the NN+ALL classifier on the UCR time series dataset. This result is not surprising, because in machine learning it is well known for a long time that ensemble classifiers often perform better than their base classifiers for reasons explained in [2]. Since any base classifier can contribute to an ensemble classifier, it is feasible to restrict comparison to base classifiers such as the state-of-the-art NN+ALL classifier.

The second key observation indicates that EL classifiers are clearly superior to NN classifiers with one prototype per class, denoted by NN$^1$ henceforth. Evidence for this finding is provided by two results: first, AHC and KME performed best among several prototype selection methods for NN classification [10]; and second, error rates of EL classifiers are significantly better than those of NN+AHC and NN+KME for eight, comparable for two, and significantly worse for two datasets.

The third key observation is that EL classifiers clearly better compromise between solution quality and computation time than NN classifiers. Findings reported by [24]
| Dataset       | ALL  | AHC   | KME   | ePERC | eLOGR | eMARG | eLSVM |
|---------------|------|-------|-------|-------|-------|-------|-------|
| Coffee        | 17.9 | 25.0  | 25.0  | 4.6 ± 3.0 | 4.5 ± 2.9 | 4.7 ± 3.3 | 3.1 ± 2.9 |
| ECG200        | 23.0 | 28.0  | 28.0  | 13.6 ± 2.1 | 11.8 ± 2.1 | 13.6 ± 2.1 | 13.1 ± 1.7 |
| ECGFiveDays   | 23.2 | 33.0  | 33.0  | 15.3 ± 2.3 | 15.7 ± 2.3 | 15.3 ± 2.4 | 11.1 ± 2.0 |
| ItalyPowDem.  | 5.0  | 21.5  | 21.5  | 3.8 ± 2.1 | 3.0 ± 2.0 | 3.5 ± 2.0 | 3.0 ± 2.0 |
| Wafer         | 0.2  | 69.5  | 69.5  | 1.3 ± 0.3 | 1.2 ± 0.2 | 1.0 ± 0.2 | 1.0 ± 0.2 |
| Gun Point     | 9.3  | 32.7  | 32.7  | 9.7 ± 2.6 | 9.2 ± 2.5 | 10.0 ± 3.4 | 9.0 ± 2.8 |
| SonyAIBO II   | 27.5 | 21.6  | 21.6  | 27.0 ± 2.3 | 20.2 ± 2.1 | 26.6 ± 3.3 | 22.7 ± 2.1 |
| Lighting 2    | 13.1 | 36.1  | 36.1  | 44.2 ± 4.2 | 44.1 ± 4.4 | 44.5 ± 4.4 | 44.6 ± 2.9 |
| MoteStrain    | 16.5 | 13.3  | 13.3  | 17.2 ± 2.6 | 16.0 ± 2.3 | 17.6 ± 2.3 | 15.8 ± 2.3 |
| SonyAIBO      | 16.9 | 18.8  | 18.8  | 19.3 ± 2.5 | 18.6 ± 2.5 | 19.5 ± 2.6 | 17.8 ± 2.4 |
| TwoLeadECG    | 9.6  | 16.2  | 16.2  | 22.7 ± 2.3 | 21.8 ± 2.4 | 21.7 ± 2.4 | 21.8 ± 2.3 |
| Yoga          | 16.4 | 45.8  | 45.8  | 20.9 ± 2.1 | 21.5 ± 1.0 | 21.1 ± 1.1 | 20.8 ± 1.1 |

Table 3: Mean error rates and standard deviation of elastic linear classifiers averaged over 100 trials and error rates of nearest-neighbor classifiers using the DTW distance (NN+DTW). ALL: NN+DTW with all training examples as prototypes; AHC: NN+DTW with one prototype per class obtained from agglomerative hierarchical clustering with Ward linkage; KME: NN+DTW with one prototype per class obtained from k-means clustering; ePERC: elastic perceptron; eLOGR: elastic logistic regression; eMARG: elastic margin perceptron; eLSVM: elastic linear SVM. Best (avg.) results are highlighted. Green rows: avg. results of all elastic linear classifiers are better than the results of all NN classifiers. Yellow rows: results of elastic linear classifiers and NN classifiers are comparable. Red rows: avg. results of all elastic linear classifiers are worse than the best result of an NN classifier.
indicate that more prototypes may improve generalization performance of NN classifiers. At the same time, more prototypes increase computation time, though the differences will decrease for larger number of prototypes by applying certain acceleration techniques. At the extreme ends of the scale, we have NN+ALL and NN classifiers. With respect to solution quality, the first key observation states that EL classifiers are comparable to the slowest NN classifiers using the whole training set as prototypes and clearly superior to the fastest NN classifiers using one prototype per class. To compare computational efficiency, we first consider the case without applying any acceleration techniques. We measure computational efficiency by the number of proximity calculations required to classify a single time series. This comparison is justified, because the complexity of computing a DTW distance and an elastic inner product are identical. Then EL classifiers are $p$-times faster than NN classifiers, where $p$ is the number of prototypes. Thus the fastest NN classifiers effectively have the same computational effort as EL classifiers for arbitrary multi-class problems, but they are not competitive to EL classifiers according to the second key observation. Next, we discuss computational efficiency of both types of classifiers, when one applies acceleration techniques. For NN classifiers, two common techniques to decrease computation time are global constraints such as the Sakoe-Chiba band \cite{21} and diminishing the number of DTW distance calculations by applying lower bounding technique \cite{19, 20}. Both techniques can equally well be applied to EL classifiers, where lower-bounding techniques need to be converted to upper-bounding techniques. Furthermore, EL classifiers can additionally control the computational effort by the number $m$ of columns of the matrix space. Here $m$ was set to 10\% of the length $n$ of the shortest time series of the training set. The better performance of EL classifiers in comparison to NN classifiers is notable, because the decision boundaries that can be implemented by their counterparts in the Euclidean space are both the set of all hyperplanes. We assume that EL classifiers outperform NN classifiers for the following reason: First, learning prototypes by clustering minimizes a cluster criterion unrelated to the risk functional of a classification problem. Therefore the resulting prototypes may fail to discriminate the data for some problems. Second, in view of Algorithm 1 the prototypes learned by AHC or KME correspond to matrices of an $n$-dimensional subspace of $\mathcal{X} = \mathbb{R}^{n \times n}$. The weight matrix learned by an elastic linear classifier is from a $(n \times m)$-dimensional subspace of $\mathcal{X}$, where $m = 0.1 \cdot n > 1$ for all datasets. This shows that EL classifiers have higher capacity than NN classifiers.

The fourth key observation is that the strong assumption of elastic-linearly separable problems is appropriate for some problems in the time series classification. Error rates of elastic linear classifiers for Coffee, ItalyPowerDemand, and Wafer are below 5\%. For these problems, the strong assumption made by EL classifiers is appropriate. For all other datasets, the high error rates of EL classifiers could be caused by two factors: first, the assumption that the data is elastic-linearly separable is inappropriate; and second, the number of training examples given the length of the time series is too low for learning (see ratio $\rho$ in Table 2). Here further experiments are required.

Finally, we observed that the different EL classifiers perform comparable with advantages for eLOGR and eLSVM. These findings correspond to similar findings for logistic regression and linear SVM in vector spaces.
To summarize, the results show that elastic linear classifiers are simple and efficient methods. They rely on the strong assumption that an elastic-linear decision boundary is appropriate. Therefore, elastic linear classifiers may yield inaccurate predictions when the assumptions are biased towards oversimplification and/or when the number of training examples is too low compared to the length of the time series. These findings are in line with those of linear classifiers in Euclidean space.

6. Conclusion

This paper introduces generalized gradient methods for learning on time series under elastic transformations. This approach combines (a) the novel concept of elastic functions that links elastic proximities on time series to piecewise smooth functions with (b) generalized gradient methods for non-smooth optimization. Using the proposed scheme, we (1) showed how a broad class of gradient-based learning can be applied to time series under elastic transformations, (2) derived general convergence statements that justify the generalizations, and (3) placed existing adaptive methods into proper context. Exemplarily, elastic logistic regression, elastic (margin) perceptron learning, and elastic linear SVM have been tested on two-class problems and compared to nearest neighbor classifiers using the DTW distance. Despite the simplicity in terms of the decision boundary and the computational efficiency, elastic linear classifiers perform convincing. There is still room for improvement by controlling the capacity via the number of columns of the weight matrix and by applying different forms of regularization. The results indicate that adaptive methods based on elastic functions may complement the state-of-the-art in statistical pattern recognition on time series, in particular when powerful non-linear gradient-based methods such as deep learning are extended to time series under elastic transformations.

References

[1] G.E. Batista, X. Wang, and E.J. Keogh. A Complexity-Invariant Distance Measure for Time Series. SIAM International Conference on Data Mining, SDM, 11:699–710, 2011.

[2] T.G. Dietterich. Ensemble methods in machine learning. Proceedings of the First International Workshop on Multiple Classifier Systems, 2000.

[3] Y. Ermoliev and V. Norkin. Stochastic generalized gradient method for nonconvex nonsmooth stochastic optimization. Cybernetics and Systems Analysis, 34(2):196–215, 1998.

[4] T. Fu. A review on time series data mining. Engineering Applications of Artificial Intelligence, 24(1):164–181, 2011.

[5] P. Geurts. Pattern extraction for time series classification. Principles of Data Mining and Knowledge Discovery, pp. 115–127, 2001.
[6] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer, 2001.

[7] V. Hautamaki, P. Nykanen, and P. Franti. Time-series clustering by approximate prototypes. *International Conference on Pattern Recognition, ICPR*, 2008.

[8] E. Keogh, Q. Zhu, B. Hu, Y. Hao., X. Xi, L. Wei, and C. A. Ratanamahatana. The UCR Time Series Classification/Clustering Homepage: [www.cs.ucr.edu/~eamonn/time_series_data/](http://www.cs.ucr.edu/~eamonn/time_series_data/), 2011.

[9] J.B. Kruskal and M. Liberman. The symmetric time-warping problem: From continuous to discrete *Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison*, p. 125–161, 1983

[10] J. Lines and A. Bagnall. Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 2014.

[11] A. Nedic and D.P. Bertsekas. Incremental subgradient methods for nondifferentiable optimization. *SIAM Journal on Optimization*, 12(1):109–138, 2001.

[12] V. Niennattrakul and C.A. Ratanamahatana. Inaccuracies of shape averaging method using dynamic time warping for time series data. *Computational Science — ICCS*, pp. 513–520, 2007.

[13] V. Niennattrakul and C.A. Ratanamahatana. On Clustering Multimedia Time Series Data Using K-Means and Dynamic Time Warping *International Conference on Multimedia and Ubiquitous Engineering*, pp. 733–738, 2007.

[14] V. Norkin. Stochastic generalized-differentiable functions in the problem of non-convex nonsmooth stochastic optimization. *Cybernetics and Systems Analysis*, 22(6):804–809, 1986.

[15] F. Petitjean, A. Ketterlin, and P. Gancarski. A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition*, 44(3): 678–693, 2011.

[16] F. Petitjean, G. Forestier, G.I. Webb, A.E. Nicholson, Y. Chen, and E. Keogh. Dynamic Time Warping Averaging of Time Series allows Faster and more Accurate Classification. *International Conference on Data Mining, ICDM*, 2014.

[17] L.R. Rabiner and J.G. Wilpon. Considerations in applying clustering techniques to speaker-independent word recognition. *The Journal of the Acoustical Society of America*, 66(3): 663–673, 1979.

[18] L.R. Rabiner and J.G. Wilpon. A simplified, robust training procedure for speaker trained, isolated word recognition systems. *The Journal of the Acoustical Society of America*, 68(5):1271–1276.
[19] C. A. Ratanamahatana and E. J. Keogh. Making time-series classification more accurate using learned constraints. *SIAM International Conference on Data Mining*, 2004.

[20] C. A. Ratanamahatana and E. J. Keogh. Three myths about dynamic time warping data mining. *SIAM International Conference on Data Mining*, pp. 506–510, 2005.

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

[22] N.Z. Shor. *Minimization Methods for Nondifferentiable Functions*. Springer-Verlag, Berlin, 1985.

[23] P. Somervuo and T. Kohonen. Self-organizing maps and learning vector quantization for feature sequences. *Neural Processing Letters*, 10(2): 151–159, 1999.

[24] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh. Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, 26(2): 275–309, 2013.

[25] J.P. Wilpon and L.R. Rabiner. A modified K-means clustering algorithm for use in isolated work recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 33(3): 587–594, 1985.

[26] X. Xi, E. Keogh, C. Shelton, L. Wei, and C.A. Ratanamahatana. Fast time series classification using numerosity reduction. *Proceedings of the 23rd International Conference on Machine Learning*, pp. 1033–1040, 2006.

A. Proof of Convergence Results for Elastic Linear Classifiers

Since affine functions are convex and the maximum of convex functions is also convex, the elastic inner product is convex. In addition, the composition of convex functions is convex. Therefore the loss functions of elastic linear classifiers are convex. Then the first convergence results is shown in [22].

To show the two other convergence statements, we assume that $|\mathcal{D}| = N$. For each training example $(x_i, y_i) \in \mathcal{D}$ the loss

$$\ell_i(\theta) = \ell_i(y_i, b + \sigma(x_i, W))$$

is real-valued and convex, where $\theta = (W, b)$. Then there is a positive scalar $C_i$ that bounds the subdifferential of $\ell_i$ at $\theta$ for all $i \in [N]$. Suppose that

$$C = \max_{i=1,\ldots,N} C_i.$$

Then from [11], Prop. 2.2. follows that the incremental generalized gradient method converges to a local minimum.
To show the Elastic Margin Perceptron Convergence Theorem, we assume that

$$E_N[\theta] = \sum_{i=1}^{N} \ell_i(\theta)$$

is the error without averaging operation, that is $E_N = N \cdot R_N$. By assumption, the training set $\mathcal{D}$ is elastic-linearly separable. Then the minimum value $E_*$ of $E_N$ is zero. From [11], Prop. 2.1. follows

$$\lim_{t \to \infty} E_N(\theta^t) \leq E_* + \frac{\eta \cdot C^2}{2} = \frac{\eta \cdot C^2}{2},$$

where $\eta$ is the learning rate. Choosing $\eta \leq \xi/C^2$ gives

$$\lim_{t \to \infty} E_N(\theta^t) \leq \frac{\xi}{2}.$$

Since $\xi > 0$, this implies that there is a $t_0$ such that $\ell_t(\theta^t) < \xi$ for all $t \geq t_0$. Here, $\ell_t$ refers to example $(x_t, y_t) \in \mathcal{D}$ presented at iteration $t$. From this follows that all training examples are classified correctly after a finite number of update steps, provided that $\lambda \leq \xi$. $\blacksquare$