Data Min Knowl Disc (2017) 31:233–263
DOI 10.1007/s10618-016-0462-1

Reliable early classification of time series based on discriminating the classes over time

Usue Mori[1] · **Alexander Mendiburu**[2] · **Eamonn Keogh**[3] · **Jose A. Lozano**[1]

Received: 8 July 2015 / Accepted: 18 April 2016 / Published online: 29 April 2016

Abstract The goal of early classification of time series is to predict the class value of a sequence early in time, when its full length is not yet available. This problem arises naturally in many contexts where the data is collected over time and the label predictions have to be made as soon as possible. In this work, a method based on probabilistic classifiers is proposed for the problem of early classification of time series. An important feature of this method is that, in its learning stage, it discovers the timestamps in which the prediction accuracy for each class begins to surpass a pre-defined threshold. This threshold is defined as a percentage of the accuracy that would be obtained if the full series were available, and it is defined by the user. The class predictions for new time series will only be made in these timestamps or later. Furthermore, when applying the model to a new time series, a class label will only be provided if the difference between the two largest predicted class probabilities is higher than or equal to a certain threshold, which is calculated in the training step. The proposal is validated on 45 benchmark time series databases and compared with several state-of-the-art methods, and obtains superior results in both earliness and accuracy. In addition, we show the practical applicability of our method for a real-

Responsible editor: Toon Calders.

✉ Usue Mori
usue.mori@ehu.eus

1 Intelligent Systems Group (ISG), Department of Computer Science and Artificial Intelligence, University of the Basque Country UPV/EHU, Manuel de Lardizabal 1, 20018 Donostia/San Sebastián, Spain

2 Intelligent Systems Group (ISG), Department of Computer Architecture and Technology, University of the Basque Country UPV/EHU, Manuel de Lardizabal 1, 20018 Donostia/San Sebastián, Spain

3 Department of Computer Science and Engineering, University of California, Riverside, CA, USA

world problem: the detection and identification of bird calls in a biodiversity survey scenario.

Keywords Time series classification · Early classification · Gaussian process

1 Introduction

In recent years, the increasing proliferation in data collecting devices has generated a new database type in which each instance consists of a time series. In order to extract useful information from these databases, many common data mining tasks have been extended and adapted to the specific characteristics of temporal data. One of the most popular tasks is time series classification (Xing et al. 2010), a supervised learning problem where the objective is to predict the class membership of time series as accurately as possible. There are many application scenarios that naturally adapt to the supervised classification of time series. Some examples are using electrocardiography (ECG) data to predict whether a patient has heart disease or not (Kadous and Sammut 2005) or sign language (Kadous and Sammut 2005) and motion (Li et al. 2006) classification using multivariate time series data.

Over the years, many extensions of the time series classification problem have been studied and early classification of time series (Xing et al. 2011a) is one of the most notable and that which will be studied in this work. This problem arises in contexts where the data is collected over time and it is preferable, or even necessary, to predict the class labels of the time series as early as possible.

As the most typical example, in medical informatics, the patient's clinical data is monitored and collected over time, and the early diagnosis of some diseases is highly correlated with a positive prognosis. For example, Evans et al. (2015) state that some indicators of physiologic deterioration such as tachypnea, tachycardia, hypotension, decreased oxygen saturation, and changes in level of consciousness are predictors of other serious medical complications in hospitalized patients. In this context, they show that the monitoring of patients and early identification of physiologic deterioration can be used to raise alerts and prevent crises. In addition, there are also many other applications apart from medical informatics in which early classification can be useful. For example, Ghalwash et al. (2014) mention early stock crisis identification. Bregón et al. (2006) apply early classification to rapidly classify different types of faults in a simulated industrial plant. Hatami and Chira (2013) attempt to classify a set of different odors as early as possible by using odor signals obtained from a set of sensors. This could be used, for example, to identify chemical spills or leaks. Finally, in Sect. 7, we show a case study related to the early identification of bird species, based on their songs, which can be used to trigger the cameras deployed in their natural habitats and make better use of their battery life.

It is easy to imagine that the earliness of the predictions may affect their accuracy. As a general rule, as more data points become available, the class predictions become more accurate because the information about the time series is more complete. In view of this, in early classification of time series, the aim is to find a trade-off between two conflicting objectives: the accuracy of the predictions and their earliness. Class predictions should be given as early as possible while maintaining a satisfactory level of accuracy (Xing et al. 2011a).

In this paper, we present an early classification framework for time series based on class discriminativeness and reliability of predictions (ECDIRE). As its name implies, in its training phase, the method analyzes the discriminativeness of the classes over time and selects a set of time instants in which the accuracy for each class begins to exceed a pre-defined threshold. This threshold is defined as a percentage of the accuracy that would be obtained if the full length series were available and its magnitude is selected by the user. The information obtained from this process will help us decide at which instants class predictions for new time series will be carried out, and will be useful to avoid premature predictions. Furthermore, ECDIRE is based on probabilistic classifiers (PC), so the reliability of the predicted class labels is controlled by using the class probabilities extracted from the classification models built in the training process.

The rest of the paper is organized as follows. In Sect. 2, we formally introduce the problem of early classification and in Sect. 3 we summarize the related work. Section 4, presents the main contribution of this paper: the ECDIRE framework. In the following section, we make a brief introduction of Gaussian Process classification models, the specific probabilistic models used in this work, and present an adaptation for the task of time series classification. In Sect. 6, the experimentation and validation of this method is carried out, and in Sect. 7, we present a case study on a real problem regarding the early identification of bird calls in a stream of recorded data. Finally, in Sect. 8, we summarize the main conclusions and propose some future research directions.

2 Early classification of time series: problem setting

A *time series* is an ordered sequence of pairs (timestamp, value) of finite length L (Xing et al. 2011a):

$$TS = \{(t_i, x_i), \;\; i = 1, ..., L\}, \tag{1}$$

where we assume that the timestamps $\{t_i\}_{i=1}^{L}$ take positive and ascending real values. The values of the time series (x_i) may be univariate or multivariate and take real values. Instead, if the $\{x_i\}_{i=1}^{L}$ take values from a finite set, it is commonly denominated a *sequence*. Finally, a *database of time series* is an unordered set of time series which can all be of the same length, or can have different lengths.

With these definitions at hand, *time series classification* is defined as a supervised data mining task where, given a training set of complete time series and their respective class labels $X = \{(TS_1, CL_1), (TS_2, CL_2), ..., (TS_n, CL_n)\}$, the objective is to build a classifier that is able to predict the class label of any new time series as accurately as possible (Xing et al. 2010, 2011a).

As a variant of time series classification, early classification of time series consists in predicting the class labels of the new time series as early as possible, preferably before the full sequence lengths are available. This problem appears in many domains in which data arrives in a streaming manner, and early class predictions are important because of the cost of collecting data or due to the costs associated with making later predictions.

The earlier a prediction is made, fewer data points from the time series will be available and, in general, it will be more difficult to provide accurate predictions. However, as mentioned previously, in early classification of time series, the class predictions should be carried out as soon as possible. In this context, the key to early classification of time series is not only to maximize accuracy, but rather to find a trade-off between two conflicting objectives: accuracy and earliness. The aim is to provide class values as early as possible while ensuring a suitable level of accuracy.

In the following sections, for the sake of simplicity in the explanations and also because the publicly available databases used in the experimentation are of this type, we will focus on datasets conformed with time series of the same length L. However, with a few changes, the ECDIRE method could also be applied to sequences that take values from a finite set, databases with series of different lengths, and even to series of unknown lengths. The necessary modifications will be commented on as we introduce the method.

3 Early classification of time series: related work

The solutions proposed in the literature for the problem of early classification of time series are varied. Some methods simply learn a model for each early timestamp and design different mechanisms to decide which predictions can be trusted and which should be discarded. To classify a new time series, predictions and reliability verifications are carried out systematically at each timestamp and the class label is issued at the first timestamp in which a reliable prediction is made. For example, in Ghalwash et al. (2012), a hybrid hidden Markov/support vector machine (HMM/SVM) model is proposed and a threshold is set on the probability of the class membership to try to ensure the reliability of the predicted labels. In Hatami and Chira (2013), an ensemble of two classifiers is proposed, which must agree on the class label in order to provide a prediction. Lastly, Parrish et al. (2013) present a method based on local quadratic discriminant analysis (QDA). In this case, the reliability is defined as the probability that the predicted class label using the truncated time series and the complete series will be the same. At each timestamp, the method checks if this probability is higher than a user pre-defined threshold and gives an answer only if this is so.

A slightly different strategy can be found in Xing et al. (2011b), Ghalwash et al. (2014) and He et al. (2015), where the authors propose different methods to discover a set of shapelets which are useful to discriminate between the classes early in time. The concept of *shapelet* was first introduced by Ye and Keogh (2009) and refers to a subsequence or subpattern of a time series that can be used to represent and identify a given class. To classify a new time series in an environment where the data is collected over time, each time a new data point arrives, the series is compared to the library of shapelets and is classified if a match is found. The matching condition depends on the required reliability level and is determined in the training process.

Finally, we must mention the early classification on time series (ECTS) method presented by Xing et al. (2011a). This work formally defines the problem of early classification for the first time, and analyzes the stability of the nearest neighbor relationships in the training set as the length of the series is truncated. In its training

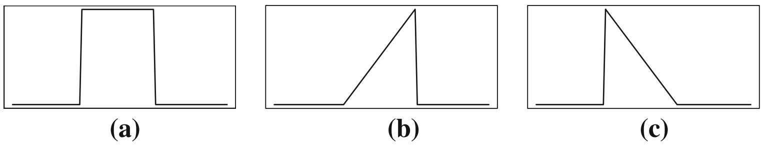

Fig. 1 The idealized shapes that define the CBF synthetic database. The actual items are distorted by random noise and warping. **a** Cylinder. **b** Bell. **c** Funnel

phase, the ECTS method calculates the minimum prediction length (MPL) for each time series in the training set. This value represents the earliest timestamp in which the given training series begins to provide reliable nearest neighbor class predictions. To predict the class label of a new unclassified time series, at each early timestamp t, all the training series are truncated to the corresponding length and a simple nearest neighbor classifier (1NN) is applied. However, once the nearest neighbor is found, if its MPL is larger than t, the prediction is considered unreliable, no class label is provided and the method simply waits until the next data point arrives.

Based on the characteristics and flaws of the methods proposed in the literature, in this work, we design an early classification method that will focus on three aspects. Firstly, note that a clear disadvantage of most of the early classification methods in the literature is that, when trying to predict the class label of a new time series, forecasts must be made and checked at all timestamps. Only the ECTS method can avoid making some predictions at the initial early instants, when the MPLs of absolutely all the training series are larger than the given timestamp. This results in many unnecessary calculations that could be avoided if the typology of the different classes were taken into account. The synthetic CBF database (Keogh et al. 2011) is a clear example of the disadvantage of issuing and checking the predictions at each timestamp. This database is identified by three shapes, each of which represents a class (see Fig. 1). The series that belong to each class are obtained by translating, stretching and compressing an initial shape and adding noise at different levels. As can be seen, the three shapes are identical until one point and, thus, it makes no sense to make predictions until this time has arrived. In order to deal with this issue, our method incorporates a mechanism to initially decide, for each class, after which time instant it makes sense to make predictions and when we can avoid unnecessary calculations.

Secondly, our method includes a procedure that controls the reliability of the class forecasts and is able to discard outlier series. We note that some proposals, such as the ECTS method, which is based on the NN classifier, simply lack a strategy to control the reliability of the issued class labels. As such, for example, if an outlier series arrives, it will inevitably be assigned to one of the classes.

Finally, even if some methods, such as Ghalwash et al. (2012), Hatami and Chira (2013), Parrish et al. (2013), Xing et al. (2011b), Ghalwash et al. (2014) and He et al. (2015) do try to control the reliability of the prediction and discard outliers, most of them are not able to provide a quantitative and interpretable measure of the uncertainty of their predictions. As pointed out in Ghalwash et al. (2014), the ability of providing such measures is a desirable property of early classification methods, because it helps the users to assess the quality of the responses they receive. Our early classification method includes a strategy to discard unreliable predictions, which directly provides an interpretable measure of the goodness of the issued predictions based on probabilities.

To sum up, although some of the existing methods deal with part of these three issues, to the extent of our knowledge, none of them propose solutions to all of them. Additionally, as will be explained in Sect. 7, our method can easily be extended to detect patterns early in streaming data, which can not be done with other early classification methods.

4 Early classification framework for time series based on class discriminativeness and reliability of predictions (ECDIRE)

As previously commented, early classification has to deal with two conflicting objectives: earliness and accuracy. In this section, we introduce the ECDIRE method, which takes these two objectives into account.

4.1 Learning phase

The learning stage of the ECDIRE method is divided into three steps. In the first step, we focus on designing a mechanism that will identify the timestamps from whence it makes sense to make forecasts, for each class. In the second step we will design a reliability condition that will control the goodness of the predictions and will directly provide a measure for the uncertainty of the predictions. Finally, once this is done, we train a set of probabilistic classifiers (PC), which will be used to issue the class predictions for the new time series.

Step 1: Analysis of the discriminativeness of the classes

In the first step of our early classification framework, we propose analyzing the database and the classes therein with the objective of deciding from which instant on it makes sense to make predictions, and when we can avoid unnecessary calculations when predicting the class label of a new time series.

We propose building a timeline that describes in which moment each class becomes "safe". This concept of safety refers to the moment in which we can begin to discriminate a certain class from all the rest of the classes with a certain degree of accuracy.

Figure 2 shows an example timeline with four relevant timestamps: t_1, t_2, t_3 and t_4. These instants are ordered in time and, at each of them, a class or set of classes becomes safe. For example, at instant t_3 we can begin to discriminate class C_5 from the rest of the classes. If we consider the classes that have appeared earlier in the timeline, we can conclude that at timestamp t_3, classes C_2, C_4, C_1 and C_5 can be predicted accurately. As will be shown in the next steps, predictions will only be provided at the timestamps that appear in the timeline or later and, by means of this mechanism, a large number of calculations can be avoided when making predictions for new time series.

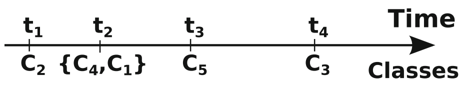

Fig. 2 An example timeline built to represent at which instant each class becomes safe

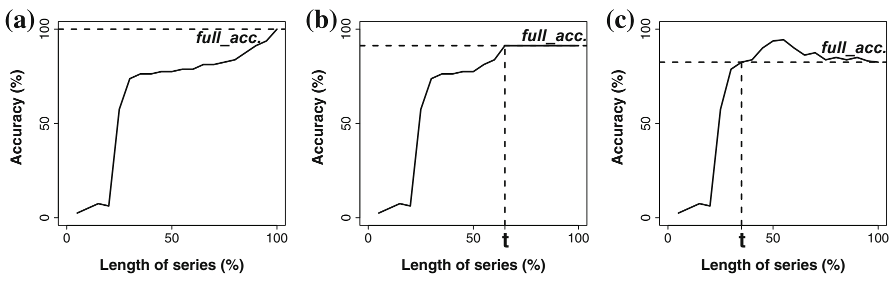

Fig. 3 Illustrative examples of the most common behaviors of the accuracy of a classifier as the number of available data points increases. The x axis refers to the percentage of the series available at the time of prediction and the y axis refers to the accuracy obtained by the classifier

The problem is now reduced to finding a method to discover the timestamps which constitute the timeline of each database. In order to do that, at each possible timestamp, a multi-class PC, which considers all the classes in the training set, is trained. As will be explained in Sect. 5.1, in this paper Gaussian Process classifiers are used, but any other classifier that provides probabilistic outputs could be used. Then, we check at which instant the accuracy becomes good enough for each class. Training a classifier at a given early time instant simply implies learning the model using only the data points collected up to that instant. We recall that in early classification problems the time series are collected over time and so, at each early timestamp, only the corresponding first points of each series are available.

To understand the intuition behind the method that we use to build this timeline, we recall that the accuracy and earliness are usually conflicting objectives (Xing et al. 2011a). In this context, typically, the higher the accuracy requirements, the later the predictions must be made. Therefore, it is common that the accuracy over time takes a shape similar to that shown in Fig. 3a. However, in practice, depending on the typology of the database and the shapes of the series therein, there are other trends that the accuracy follows frequently. For example, in some cases, if the classes are easily discriminable after a certain point, the improvement in accuracy becomes slower and tends to stabilize (see Fig. 3b). On some other occasions, the accuracy might even drop slightly after a point due to noise that the additional data may introduce (see Fig. 3c). In these two last cases, it is possible to classify the time series early in time without downgrading (or even improving) the accuracy that would be obtained if the full series were available ($full\_acc.$). The accuracy could also be strictly decreasing, constant or could take any other possible shape, although this is more unusual because there is usually some degree of conflict between both objectives (accuracy and earliness).

Based on this intuition, a straightforward strategy to build the timeline could consist in simply finding the first point in which the chosen reliability threshold is fulfilled (in the figures $full\_acc.$). However, in practice, the behavior of the accuracy tends to be noisy, with peaks and outliers that disturb the perfect trends shown in the figures. So, choosing the first point in which the accuracy threshold is fulfilled could result in overfitting. In this context, it is advisable to require some sort of stability in the behavior of the accuracy. In the following paragraphs the procedure followed to build the timeline is explained in more detail.

As we have mentioned, in a first step, a multi-class PC is trained within a stratified 10 times repeated 5-fold cross validation framework (10×5 cross validation, from now on), following the recommendations of Rodríguez et al. (2013). This is done by using the full length time series and considering all the classes. Once this is carried out, the accuracy of the model is estimated separately and locally within each class. Subsequently, the classes that yield an accuracy lower than $1/|C|$ are identified, C being the set of different classes in the database. If a random classifier, which assumes that the classes are equally distributed, were applied to the training data, the accuracy that would be obtained within each class would tend to $1/|C|$. In this context, the predictions obtained for the identified classes are worse than those that would be obtained by the random classifier. If we consider this random classifier the baseline, these class values can not be predicted with enough accuracy, even when the whole series is available. Because of this, their corresponding labels will never be given as a prediction in our early classification framework.

Of course, if desired, this condition could be modified or removed by the user. For example, note that, if the classes are unbalanced, the baseline presented above is stricter with smaller classes than with larger ones. In this case, the class frequencies in the training set could be used to define a different baseline accuracy value for each class. Of course, this could result in very low baselines for some classes, which might result in more unreliable class predictions. In this sense, one could also consider $1/|C|$ a very low baseline and require higher accuracy values for all classes. Finally, these baseline accuracy values could also be set manually using some sort of domain knowledge or based on domain specific demands.

For the rest of the classes (c), a percentage ($perc\_acc_c$) of the accuracy obtained with the full length series ($full\_acc_c$) is chosen and, with this, the desired level of accuracy is defined for each class as $acc_c = perc\_acc_c * full\_acc_c$. The goal will be to preserve this accuracy while making early predictions. With this information, the timeline can be built by following these steps:

1. Select a set of timestamps. In our case, since we must deal with many databases of which we have no domain knowledge, the timestamps in E are defined as a sequence of equidistant percentages of the length of the series in the database: $E = \{e_1\,\%, e_2\,\%, e_3\,\%, \ldots, 100\,\%\}$. Nevertheless, the user could choose a specific subset of non-equidistant percentages, based on domain knowledge or other information of the shape of the series. Additionally, if the lengths of the series are different from each other and unknown, these values would have to be substituted by the number of data-points collected from the beginning of the series, or the time passed from the first timestamp. This last choice seems more adequate, for example, for the cases in which the collection frequency of the series is different.
2. For each early timestamp $e_i \in E$:
   (a) Truncate the time series in the training set to the length specified by e_i, starting from the first point.
   (b) Train an early PC within a stratified 10x5-fold cross validation framework using these truncated sequences. The method followed to train these classifiers is explained in detail Sect. 5.

(c) For each class (c), save the mean of the accuracies obtained in all the repetitions and folds ($acc_{e_i,c}$). For a more conservative solution, choose the minimum or a low percentile of the accuracies obtained in all the repetitions and folds.

3. For each class c, choose the timestamp $e^*_c \in E$ as:

$$e^*_c = \min\{e_i \in E \ \mid \ \forall j \geq i \ \ , \ \ acc_{e_j,c} \geq acc_c\} \tag{2}$$

These e^*_c values are the timestamps that appear in the timeline and will be crucial to avoid useless predictions because, as will be shown in the next sections, forecasts will only be made at these instants or later. Please, note that, if the accuracy for a class strictly increases, then only the last timestamp in E (100 %) will fulfill the condition in Eq. 2. On the contrary, in the strictly decreasing accuracy case, all the timestamps in E will fulfill the condition so, e^*_c will be the first timestamp in E.

The timeline provides valuable information about each class and, thus, incorporating it into the ECDIRE method improves the interpretability of the model, which is a desirable property in early classification (Ghalwash et al. 2014). Finally, as we will explain in Sect. 7, the timeline obtained in this first step provides the means to directly apply the ECDIRE method to another scenario: the early detection of significant patterns in streams of unknown length (possibly infinite).

Step 2: Prediction reliability

In this step, we concentrate on the second crucial aspect of our method: designing a mechanism to control the reliability of the class predictions issued by the classifiers. As commented previously, in this work we use classifiers which provide probabilistic outputs. These values can be used to ensure the reliability of the class forecasts and to provide a quantitative measure of the quality of our predictions. The idea is to set a threshold to the differences between the probabilities assigned to the different classes with the aim of requiring a sufficiently large difference between the predicted class and the rest of the classes.

A separate threshold ($\theta_{t,c}$) is defined for each class (c) and early timestamp (t) and, for this, we use the set of classifiers built within the 10×5-fold cross validation framework presented in the 2(b) item of the previous step. We recall that each classifier corresponds to one early timestamp and, since a different threshold is sought for each early time instant, we analyze each one separately.

First, the predicted class probabilities of all the correctly classified test instances are extracted from all the folds and repetitions and they are grouped depending on the class of the test instance. Within each group, the differences between the predicted class probability of the pertinent class and the rest of the classes will be computed and the minimum of these differences will be saved for each instance. These values represent the distance in terms of differences in class probabilities of the winning class and the next most probable class for all the test instances in the group. Finally, to define the threshold for the given early timestamp and class, the minimum of all these values is taken within each group:

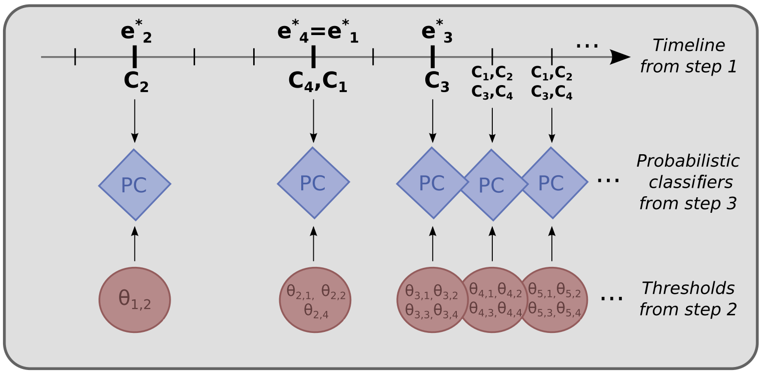

Fig. 4 An example of the ECDIRE model obtained from the three steps that conform the learning phase

$$\theta_{t,c} = \min_{x \in X_c} \left(p^t_{1:k}(x) - p^t_{2:k}(x)\right), \tag{3}$$

where $p^t_{1:k}(x)$ and $p^t_{2:k}(x)$ are the largest and second largest probability values obtained for series x at time t, k is the number of classes, and X_c is the set of correctly classified series from class c collected from all folds and repetitions.

It must be noted that choosing the minimum in this last step may lead to a very loose threshold, that will only discard very uncertain predictions, because this value is susceptible to outlier values obtained in the posterior probabilities of the training instances. For a more conservative threshold, the mean or median could be chosen by the user in this last step (see example in Sect. 7), or any other value that the user considers appropriate for the problem at hand.

Step 3: Training the set of classifiers

In this last step, we will train a set of PCs, which will later be used to classify new time series. To begin with, a classifier is built for each timestamp that appears in the timeline constructed in the previous section. Contrary to Step 1, where the PCs are learnt within a 10×5-fold cross validation scheme, in this case, the models are built by using all the training instances. Additionally, if the last timestamp in the timeline (e^*_k) does not coincide with 100% of the length of the series, the ensemble of classifiers will be complemented with classifiers built at all the posterior timestamps in E, namely in $T = \{e_i \in E : (e_i > e^*_k)\}$. This means that, after timestamp e^*_k, our method becomes similar to those in the literature but, until this point, many unnecessary models will be discarded. As for the models built in Step 1, in this case, all the classifiers will also consider all the classes in the database.

In Algorithm 1, the pseudo-code for the entire learning step is provided. Additionally, in Fig. 4, an example of the ECDIRE model obtained for a database with four classes is represented. In this case, the last timestamp in the timeline is e^*_3. As can be seen, PCs are only available at the timestamps that appear in the timeline, and later. Furthermore, in each case, only the classes which are safe are considered.

Algorithm 1 Pseudo-code for the learning phase of the ECDIRE method.

```
Input:
Training set: $X = \{(TS_1, CL_1), (TS_2, CL_2), ..., (TS_n, CL_n)\}$
Number of classes: numclass
Grid of timestamps to consider: $E = \{e_1, e_2, ..., e_k\}$
Percentage of accuracies for each class: $perc\_acc_c, \ \forall c$

1: for $e_i$ in $E$ do
2:   # 10x5 cross validation procedure
3:   for repetition in 1:10 do
4:     for fold in 1:5 do
5:       # Build training and testing sets
6:       traincv$\leftarrow$training set for repetition and fold.
7:       testcv$\leftarrow$ testing set for given repetition and fold.
8:       # Train PC model
9:       model$\leftarrow$trainpcmodel(traincv, $e_i$)
10:      # Obtain predicted classes and posterior probabilities for testing set.
11:      predicted.classes$\leftarrow$predict.classes(model, testcv)
12:      posterior.probabilities$\leftarrow$predict.probabilities(model, testcv) #One line per test instance
13:      for c in 1:numclass do
14:        $acc\leftarrow$intraclass accuracy for class $c$
15:        Save $acc$ in vector $accvector_{e_i,c}$
16:        $probs\leftarrow$posterior probabilities of correctly classified series from class $c$
17:        Save $probs$ in matrix $probs_{e_i,c}$
18:      end for
19:    end for
20:  end for
21: end for
22: # Calculate reliability threshold (Step 1)
23: for c in 1:numclass do
24:   #Calculate mean accuracies.
25:   for $e_i$ in E do
26:     $acc_{e_i,c}\leftarrow mean(accvector_{e_i,c})$
27:   end for
28:   $acc_c \leftarrow perc\_acc_c \cdot acc_{100\%,c}$
29:   $e_c^*\leftarrow$Find timestamp that fulfills equation 2.
30:   Add $(e_c^*, c)$ to list $timeline$ (each position has two components $timestamp$ and $classes$)
31: end for
32: Sort $timeline$ by timestamp (ascending).
33: # Calculate reliability threshold (Step 2)
34: for $e_i$ in E do
35:   $\theta_{e_i,c}\leftarrow$min(row.max($probs_{e_i,c}$)-row.second.max($probs_{e_i,c}$))
36: end for
37: # Train GP classifiers from Step 3
38: for t in 1:length($timeline$) do
39:   $model \leftarrow$ trainpcmodel(X, timeline.timestamps[t])
40:   Save $model$ into list $pcmodels$
41: end for
42: for $t \in E$ where $t >$ timeline.timestamps[length($timeline$)] do
43:   model$\leftarrow$trainpcmodel(X, t)
44:   Save $model$ to list $pcmodels$
45:   Save $(t, \{c_1, c_2, ..., c_k\})$ to $timeline$
46: end for
47: return $pcmodels, timeline, \Theta$
```

4.2 Prediction phase

In this phase, the objective is to make early class predictions for new unclassified time series. As commented previously, PC models are only available at the timestamps that appear in the timeline obtained in Step 1 and at all the timestamps after the last point in the timeline, so predictions will only be made at these time instants. Furthermore, although the classifiers are built including all the classes, only those that are safe will be considered at each instant. If the classifier assigns a non-safe label to a given test series, the answer will be ignored and the time series will continue in the process and wait until the next time step, when more data will be available.

Moreover, the series that are assigned to a safe class must pass the test of the prediction reliability. The differences between the predicted class probabilities obtained from the classification will be compared to the thresholds extracted in Step 2. Only if they are larger, will a final class be given. If not, the series will not be classified and will continue in the process and wait until enough new data points are available.

Finally, the class probabilities obtained for the classified test instances can be used as an estimate of the uncertainty of the prediction. As commented previously and as stated by Ghalwash et al. (2014), the availability of measures of this type is a desirable property for early time series classification problems, and providing these values may be useful and informative for users.

As a summary of this process, the pseudo-code of the prediction phase for ECDIRE is shown in Algorithm 2.

Algorithm 2 Pseudo-code for the prediction phase of the ECDIRE method.

```
Input:
$TS$: new time series
$pcmodels$, $timeline$, $\Theta$

1: for t in 1:length($timeline$) do
2:   # Predict class and posterior probabilities.
3:   predicted.class←predict.classes(pcmodels[t],TS)
4:   if (predicted.class is in timeline.classes[t]) then
5:     posterior.probabilities←predict.probabilities(pcmodels[t], TS)
6:     diff←max(posterior.probabilities)-second.max(posterior.probabilities)
7:     if diff $\geq \theta_{t,predicted.class}$ then
8:       return (predicted.class, posterior.probabilities)
9:     end if
10:  end if
11:  return (NA,NA)
12: end for
```

5 Probabilistic classifiers in the ECDIRE framework

Based on the general design of our early classification framework, any classification algorithm with probabilistic outputs could be used to build the classifiers in Sect. 4.1. In this case, we have chosen Gaussian Process (GP) models.

GP models are probabilistic models and, unlike other kernel based models such as SVMs, they output fully probabilistic predictions, which are directly applicable in our framework. Although they have been extensively applied in many domains, they are not commonly used in time series classification and, to the extent of our knowledge, have never been applied to the task of early classification. However, these models are well known for their ability to obtain good generalization results in the presence of few labeled instances (Stathopoulos et al. 2014), which is common when working with time series. Also regarding this, when working with GP models, the parameters of the kernel function can be learnt from the data within the Bayesian inference framework, whilst in SVMs this is generally done by using cross validation. Finally, in some cases, these models have shown superior performance for time series classification compared to other kernel based models such as SVMs (Stathopoulos et al. 2014).

5.1 Gaussian Process classification

A Gaussian Process (GP) (Rasmussen and Williams 2006) is an infinite collection of random variables that, when restricted to any finite set of dimensions, follows a joint multivariate Gaussian distribution. A GP is completely defined by a mean function $m(x)$ and a covariance function $k(x, x')$:

$$f(x) \sim GP(m(x), k(x, x')) \tag{4}$$

In machine learning, GPs are used as priors over the space of functions to deal with tasks such as regression and classification by applying Bayesian inference. The idea is to calculate the posterior probability of the classification or regression function, departing from a GP prior and applying Bayes' rule.

In this paper, we are interested in the use of GPs to solve classification tasks. The idea is similar to that of the logistic or probit regression. First, a continuous latent function f with a GP prior is introduced, which is a linear combination of the input variables. This function is then composed with a "link" function (σ) such as the logistic or probit function which transfers the values to the [0, 1] interval. The resultant function π is the function of interest, and a prediction for a given test instance can be obtained by calculating (Rasmussen and Williams 2006):

$$\bar{\pi}_* = p(y_* = +1|X, \mathbf{y}, \mathbf{x}_*) = \int \sigma(f_*)p(f_*|X, \mathbf{y}, \mathbf{x}_*)df_* \ , \tag{5}$$

where

$$p(f_*|X, \mathbf{y}, \mathbf{x}_*) = \int p(f_*|X, \mathbf{x}_*, \mathbf{f})p(\mathbf{f}|X, \mathbf{y})d\mathbf{f}, \tag{6}$$

and f_* is the value of the latent variable for the test instance and $\{\mathbf{x}_*, y_*\}$ and $\{X, \mathbf{y}\}$ are the input and class values of the test instance and training instances respectively. Contrary to the case of regression, this problem is not analytically tractable because the class variables are discrete and a non-Gaussian likelihood appears when calculating the posterior of the latent variable ($p(\mathbf{f}|X, \mathbf{y})$). In any case, many methods have been proposed to find a solution to this inference problem, and some examples and relevant references can be found in Rasmussen and Williams (2006).

Specifically the proposal shown in Girolami and Rogers (2006), which is valid for both binary and multinomial classification, has been applied in this work to train the GP models. Departing from a probit regression model, these authors apply a data augmentation strategy, combined with a variational and sparse Bayesian approximation of the full posterior. They first introduce a set of m_k variables, similar to the f introduced above but associated to each one of the possible class values k. Additionally, they augment the model with a set of auxiliary latent variables $y_k = m_k + \epsilon$, where $\epsilon \sim N(0, 1)$. In probit regression, this strategy enables an exact Bayesian analysis by means of Gibbs sampling, but the authors additionally propose a variational and sparse approximation of the posterior which improves the efficiency of the method. The details of this numerical approximation will not be specified in this paper due to lack of space but can be studied in the cited reference (Girolami and Rogers 2006).

5.2 Gaussian Process for time series classification

A straightforward way to apply GP classification to temporal data is to include the raw time series directly into the model as input vectors. In this case, the covariance function could be defined by any commonly used kernel function [see (Rasmussen and Williams 2006)]. The main drawback of this approach is that these common kernels are not specifically designed for time series and, as such, they are not designed to deal with shifts, warps in the time axis, noise or outliers, features that are commonly present in temporal databases. Indeed, many of these kernels are based on the Euclidean distance, and it has been proven in the past few years that this distance measure does generally not provide the best results when working with time series databases (Wang et al. 2012).

In view of this and as shown in Pree et al. (2014) and Wang et al. (2012), it seems interesting to be able to include different time series distance measures into the classification framework. A simple way of including different distance measures into a GP classification framework is to directly replace the Euclidean distance that appears in other common kernel functions by the distance measure of interest (Pree et al. 2014). However, this solution is problematic for many time series distance measures because the resultant kernel function does not necessarily create positive semi-definite covariance matrices, which are a requirement when working with GP models (Rasmussen and Williams 2006). Thus, to deal with this issue, we propose the methodology shown in Fig. 5.

In a preliminary step, a distance matrix is calculated by using any distance d of interest. Each position of this matrix holds the distance (d_{ij}) between two series TS_i and TS_j. This new feature matrix will be the input to the GP classification framework and, now, any typical kernel function may be used and the problem of non-positive semi-definite covariance matrices will not be encountered. Additionally, this procedure also allows us to combine different distance measures by concatenating the distance matrices obtained from each of them (Kate 2015) or by combining them using some operation such as a weighted sum, used in Stathopoulos et al. (2014).

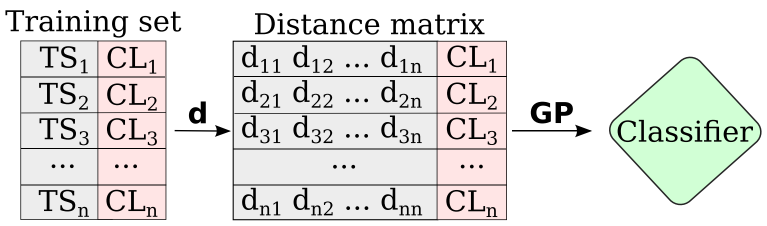

Fig. 5 Classification framework

This idea of building classifiers by using distance matrices was first introduced by Graepel et al. (1998) and can be used to easily introduce different distance measures into common classification models. Some applications of this methodology to time series databases can be found in Stathopoulos et al. (2014) and Kate (2015), where the popular dynamic time warping (DTW) distance is included into GP models and SVMs, respectively.

Additionally, note that this trick also enables training probabilistic classifiers with series of different lengths and also with series that take values from a finite set. We only have to choose a suitable distance measure which is able to deal with time series of these characteristics. As such, it allows extending the ECDIRE method to these types of databases and series quite directly.

6 Experiments

In this section, the performance of the ECDIRE method is evaluated in terms of accuracy and earliness. The complete code of the experimentation is readily available in our web page.[1]

6.1 The data

The UCR archive (Keogh et al. 2011) collects the great majority of the publicly available synthetic and real time series databases and, since its creation, has provided the baseline for evaluating new classification, clustering and indexing proposals. Some datasets from this archive have also been used to validate most previous early time series classification proposals. In this paper, the 45 time series databases from the UCR archive available at the time of the experimentation are used for validation.

6.2 Parameters of the early classification framework

The early classification framework presented in this paper requires the definition of two parameters. First of all, the number of early timestamps that ECDIRE considers have to be chosen. In this case, since the length of the series in one of the databases of UCR

[1] http://www.sc.ehu.es/ccwbayes/members/umori/ECDIRE/ECDIRE.html.

is of only 24 data points, we have chosen $E = \{5\,\%, 10\,\%, 15\,\%, \ldots, 85\,\%, 90\,\%, 95\,\%, 100\,\%\}$ in order to take a small enough granularity while avoiding repeated timestamps in E. However, for databases with longer time series, a smaller granularity could be chosen. A small set of experiments using different granularities has been included in our website[2] as supplementary material, which leads us to conclude that, in general terms, as we lower the granularity we are able to obtain earlier predictions, while maintaining the same accuracy requirements.

Additionally, recall that the desired level of accuracy (acc_c) must be tuned by the user. This parameter was defined as a percentage of the full accuracy ($full\_acc_c$) for each class. In this case, 100 % of $full\_acc_c$ is chosen for all classes. This decision has been made because the objective of many classification methods such as ECTS or RelClass is to provide classifications as early as possible but without downgrading the accuracy that would be obtained if the whole series were available. However, lower accuracy values could be chosen by the user. In this line, a small set of experiments has been included as supplementary material in our website setting $perc\_acc_c$ to 80, 85, 90 and 95 %.

6.3 Parameters and implementation of the GP models

As commented in Sect. 5.1, the GP classifiers in the ECDIRE method have been trained following the method proposed in Girolami and Rogers (2006) and, specifically, based on its implementation available in the *vbmp* package of R (Lama and Girolami 2014). This package is designed for multinomial GP classification and only works with datasets that have more than 2 classes. In view of this, we have completed the code in order to also enable binary classification by following the formulation shown by Girolami and Rogers (2006). After a small set of preliminary experiments using a set of synthetic databases, the inner product kernel has been chosen as the covariance function. This selection is by no means optimal, but the analysis of different kernel functions has been postponed for future work. In this case, the hyperparameter of the kernel function is a scaling factor which assigns a weight to each of the dimensions of the input data. We recall that each dimension of the input distance matrix corresponds to one of the series in the training set (see Fig. 5), so the hyperparameter has been fixed to a uninformative vector of ones, that gives equal importance to all the dimensions. All the remaining parameters for the *vbmp* function have been set to their default values.

The distance measure used to built the input matrix is another parameter to be chosen when building the GP classifiers. As explained in Sect. 5.2, any time series distance could be used. Nevertheless, the evaluation of the performance of the different time series similarity measures is not the objective of this paper and, with this in mind, the basic Euclidean distance has been used in the experiments presented.

[2] http://www.sc.ehu.es/ccwbayes/members/umori/ECDIRE/parameters.

6.4 Comparison to other early classification methods

In order to validate our proposal, we have compared its performance with the 1NN algorithm which uses the full length series and 3 state-of-the-art early classification methods with available source codes.[3,4]

The first early classification method is the ECTS method (Xing et al. 2011a). We compare our method to two variants of this proposal, the strict and the loose version, setting the only parameter (*minimumSupport*) to values 0, 0.05, 0.1, 0.2, 0.4 and 0.8. The second method used for comparison is the Early Distinctive Shapelet Classification (EDSC) method (Xing et al. 2011b). The authors propose two variants that show similar results in their experimentation but, in this work, only the one based on the Chebyshev Inequality is considered because of code availability. The bound for the Chebyshev condition is set to 2.5, 3 and 3.5, since the authors report that the best values are obtained within this interval. The third method is the reliable early classification method (RelClass) proposed by Parrish et al. (2013). The values for the reliability threshold τ are taken from the experimentation shown by the authors, which is also based on the datasets from the UCR: 0.001, 0.1, 0.5 and 0.9. For each of these parameters, the *Naive Gaussian Quadratic set* method and the *Gaussian Naive Bayes box* method (default method) are both used separately to calculate the reliability threshold. Furthermore, the local discriminative Gaussian dimensionality reduction has been enabled because it reduces computational costs and can reduce noise and yield higher accuracy values (Parrish et al. 2013), and the joint Gaussian estimation method has been chosen because it is more efficient and obtains similar results to those obtained by the other estimation methods considered.

6.5 Evaluation method

The databases from the UCR archive used in this study are provided with pre-specified training and testing sets. In order to enable reproduction of the results, it is common in the literature to directly use these training and testing sets in the experimentation and, in view of this, these sets have also been respected in this work. As such, the evaluation of the classification framework has been done following a train/test validation methodology. However, some supplementary experiments have been carried out using 10x5-fold cross validation on some datasets, to ensure that the results obtained by our method are not specific to these specific train/test splits, and can be accessed in our web page.

Two evaluation measures are given, each one corresponding to one of the conflicting objectives of the early classification problem. The accuracy is calculated in terms of the percentage of the correctly classified test instances:

[3] **ECTS and EDCS:** http://zhengzhengxing.blogspot.com.es/p/research.html.

[4] **Rel.Class**: http://www.mayagupta.org/publications/Early_Classification_For_Web.zip.

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} I(\hat{CL}_i = CL_i) \tag{7}$$

C_i being the true class value for test instance i and $\hat{C}_i$ its predicted value. $I(\cdot)$ takes a value of 1 if the condition is true, and 0 otherwise.

The earliness is measured by calculating the mean of the time instances t_i in which the test examples are classified. In order to give a more interpretable result, this value is normalized by the full length of the series in the database (L) and given as a percentage:

$$Earliness = \frac{1}{N}\sum_{i=1}^{N} \frac{t_i}{L} \times 100 \tag{8}$$

All the early classification methods are evaluated by using these two measures. However, in the case of the full 1NN method, as it always uses the full length of the series, we only provide the metric regarding the accuracy.

Apart from giving the raw values of these two performance measures, in order to assess the statistical differences between the different methods in each objective, we perform a set of statistical tests for each objective. First, the Friedman test is applied to conclude if there are any overall statistical differences and to obtain a ranking of the methods. Then, the Holm posthoc procedure (Demšar 2006) is applied to make pairwise comparisons between the methods in each objective. The results will show if there are statistically significant differences between each pair of methods in each of the objectives separately. This will enable us to analyze if any of the methods leans notably towards one of the objectives, or if any of them obtains particularly bad results on one of the evaluation measures. All statistical tests have been carried out using the *scmamp* package from R (Calvo and Santafé 2015), and setting the significance level to $\alpha = 0.05$.

Finally, in order to validate our proposal and compare it to the other state-of-the-art early classification methods, we must necessarily take the multi-objective nature of the early classification problem into account. The Pareto optimality criterion states that a solution dominates another solution if it obtains better results in at least one of the objectives while not degrading any of the others. Based on this criterion, by counting the times in which our method dominates the others and vice versa, we provide a multiobjective comparison of the ECDIRE method with the rest of the early classification methods considered in this work. Additionally, we also provide some further domination results, which compare ECDIRE against all the rest of the methods together.

6.6 Results

In Tables 1 and 2 the accuracy and earliness results obtained from the experimentation can be observed respectively. The results issued from Friedman's test and Holm's method are summarized in Figs. 6 and 7, and the p-values of all the performed tests

Table 1 Accuracy values for ECDIRE, ECTS, EDSC, RelClass and the full 1NN method

Dataset	ECDIRE	RelClass	ECTS	EDSC	1NN
50words	0.53 (99.12 %)	**0.66**	0.57	0.48	0.61
Adiac	0.55 (98.98 %)	**0.63**	0.40	0.16	0.61
Beef	0.50 (90 %)	**0.57**	0.50	0.23	0.53
CBF	**0.89** (100 %)	0.64	0.85	0.84	0.85
Chlorine concentration	0.56 (98.44 %)	**0.82**	0.62	0.52	0.65
CinC_ECG_torso	0.81 (99.78 %)	0.85	0.87	0.55	**0.90**
Coffee	**0.96** (100 %)	0.89	0.75	0.75	0.75
Cricket_X	0.57 (99.74 %)	**0.61**	0.56	0.52	0.57
Cricket_Y	0.63 (100 %)	**0.68**	0.63	0.57	0.64
Cricket_Z	0.60 (100 %)	**0.66**	0.59	0.0	0.62
Diatom size reduction	0.80 (92.48 %)	**0.94**	0.8	0.85	0.93
ECG200	**0.91** (100 %)	0.89	0.89	0.85	0.88
ECG five days	0.60 (100 %)	0.52	0.62	0.74	**0.80**
Face All	**0.87** (99.88 %)	0.69	0.76	0.66	0.71
FaceFour	0.61 (76.14 %)	**0.83**	0.82	0.75	0.78
FacesUCR	0.74 (99.17 %)	**0.77**	0.71	0.63	**0.77**
Fish	**0.81** (100 %)	0.79	0.75	0.68	0.78
Gun_Point	0.87 (100 %)	0.91	0.87	**0.94**	0.91
Haptics	**0.44** (91.56 %)	0.41	0.37	0.34	0.37
Inline skate	0.26 (95.09 %)	**0.27**	0.33	0.18	0.34
Italy power semand	0.93 (100 %)	0.85	0.94	0.82	**0.96**
Lighting2	0.54 (88.52 %)	0.62	0.70	**0.80**	0.75
Lighting7	0.48 (97.26 %)	**0.68**	0.58	0.67	0.58
MALLAT	0.78 (100 %)	0.73	0.85	0.59	**0.91**
Medical images	**0.74** (100 %)	0.67	0.68	0.60	0.68
Mote strain	0.80 (100 %)	0.58	**0.88**	0.78	**0.88**
Olive oil	0.40 (100 %)	0.77	**0.90**	0.6	0.87
OSU leaf	0.52 (100 %)	0.48	0.49	**0.56**	0.52
Sony AIBO robot surface	**0.83** (98.84 %)	0.79	0.69	0.80	0.70
Sony AIBO robot surfaceII	0.74 (100 %)	**0.88**	0.85	0.81	0.86
Star light curves	**0.95** (100 %)	**0.95**	0.15	–	0.85
Swedh leaf	**0.87** (99.84 %)	0.83	0.78	0.47	0.79
Symbols	0.81 (90.15 %)	0.71	0.81	0.51	**0.90**
Synthetic_control	0.96 (100 %)	**0.98**	0.88	0.89	0.88
Trace	0.77 (100 %)	**0.86**	0.74	0.80	0.76
TwoLeadECG	0.81 (100 %)	0.72	0.73	**0.88**	0.75
Two_Patterns	0.87 (99.98 %)	**0.93**	0.86	0.80	0.91
uWaveGestureLibrary_X	**0.77** (99.97 %)	0.75	0.73	0.54	0.74
uWaveGestureLibrary_Y	**0.70** (99.92 %)	0.68	0.63	0.37	0.66
uWaveGestureLibrary_Z	**0.71** (99.97 %)	**0.71**	0.65	0.52	0.65

Table 1 continued

Dataset	ECDIRE	RelClass	ECTS	EDSC	1NN
Wafer	0.97 (100 %)	0.99	0.99	0.99	**1.00**
Words synonyms	0.52 (98.43 %)	**0.65**	0.59	0.47	0.62
Yoga	**0.85** (99.9 %)	0.83	0.81	0.71	0.83
NIFECG_Thorax1	**0.89** (99.85 %)	0.87	0.81	–	0.83
NIFECG_Thorax2	**0.93** (99.85 %)	0.88	0.88	–	0.88

For the ECDIRE method the percentage of classified series are included between parentheses. The method/s with the highest accuracy in each database are shown in bold

Table 2 Earliness values for ECDIRE, ECTS, EDSC and RelClass

Dataset	ECDIRE	RelClass	ECTS	EDSC
50 words	**40.30**	92.20	72.86	58.89
Adiac	**38.54**	96.04	59.09	84.55
Beef	67.78	**25.70**	76.5	93.61
CBF	28.55	**23.08**	71.5	31.85
Chlorine concentration	**14.42**	97.59	66.07	33.33
CinC_ECG_torso	49.71	56.58	0.58	**43.63**
Coffee	82.14	**38.44**	83.94	54.23
Cricket_X	**47.98**	78.68	71.80	52.57
Cricket_Y	**36.00**	82.36	66.49	45.10
Cricket_Z	**45.99**	80.36	67.86	56.12
Diatom size reduction	24.26	33.49	**14.88**	27.04
ECG200	90.10	68.81	60.11	**23.24**
ECG five days	21.07	**15.84**	63.82	53.60
Face all	56.49	96.27	63.85	**38.94**
Face four	**22.31**	34.22	72.26	47.98
FacesUCR	59.15	92.71	87.21	**51.58**
Fish	55.17	85.42	60.94	**47.70**
Gun_Point	**32.37**	71.33	46.92	45.58
Haptics	86.52	57.89	93.87	**12.53**
Inline skate	**33.83**	87.31	85.08	46.69
Italy power demand	70.16	**35.92**	79.33	67.08
Lighting2	**09.07**	61.16	89.01	55.14
Lighting7	**19.93**	85.23	86.97	68.40
MALLAT	45.35	44.01	69.32	**39.96**
Medical images	**21.20**	88.96	53.87	31.95
Mote strain	**12.10**	90.94	79.06	38.08
Olive oil	30.00	**18.76**	87.34	38.82
OSU leaf	**47.52**	97.10	76.59	54.38
Sony AIBO robot surface	62.26	57.70	68.49	**47.03**

Table 2 continued

Dataset	ECDIRE	RelClass	ECTS	EDSC
Sony AIBO robot surface II	**17.66**	70.86	54.54	35.51
Star light curves	**53.10**	90.02	82.25	–
Swedh leaf	**45.97**	91.96	76.27	62.34
Symbols	**45.33**	45.82	51.30	60.25
Synthetic_control	61.92	71.54	87.88	**50.81**
Trace	41.75	77.82	50.72	**38.63**
Two lead ECG	69.38	83.63	64.43	**46.85**
Two_Patterns	98.76	91.82	86.52	**64.04**
uWaveGestureLibrary_X	74.03	90.09	85.98	**64.30**
uWaveGestureLibrary_Y	97.09	81.96	86.29	**70.14**
uWaveGestureLibrary_Z	75.56	91.80	85.03	**61.18**
Wafer	**10.87**	30.75	44.38	27.99
Words synonyms	66.23	91.4	82.51	**65.66**
Yoga	100.0	87.28	69.41	**38.57**
NIFECG_Thorax1	**64.33**	93.47	78.22	–
NIFECG_Thorax2	**56.58**	92.16	76.84	–

The method/s with the lowest earliness value in each database are shown in bold

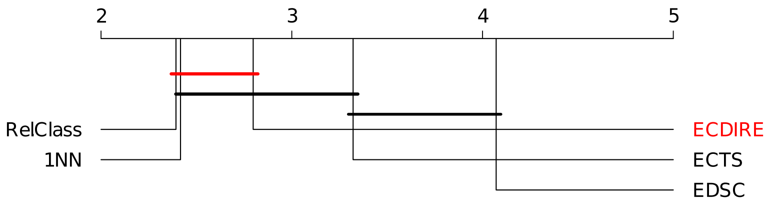

Fig. 6 Results from statistical tests for accuracy. The ranking is issued from the Friedman test and shows the goodness of the methods from better to worse (*left* to *right*). The *bold lines* show the results issued from the Holm procedure, and joins pairs of methods that do not yield statistically significant differences

can be found in our website.[5] Note that, for the comparison methods, we only show the results for the parameter configuration that dominates our method most times. If there are ties within a method, the configuration that is dominated a lower number of times by our method is chosen. Following this procedure, the following parameter combinations have been selected: RelClass with $\tau = 0.5$ and the Gaussian Naive Bayes box method, the loose version of ECTS with the support set to 0, and the EDSC method with a threshold value of 2.5. In any case, the raw results for all the rest of parameter configurations can be found in our website.[6]

Note that the results for the "StarlightCurves", "NIFECG_Thorax1" and "NIFECG _Thorax2" databases are ommited for the EDSC method in both tables (accordingly,

[5] http://www.sc.ehu.es/ccwbayes/members/umori/ECDIRE/pvalues.

[6] http://www.sc.ehu.es/ccwbayes/members/umori/ECDIRE/ECDIRE.html.

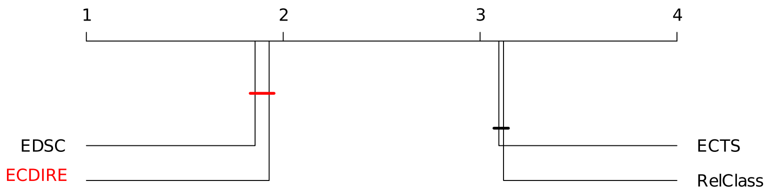

Fig. 7 Results from statistical tests for earliness. The ranking is issued from the Friedman test and shows the goodness of the methods from better to worse (*left* to *right*). The *bold lines* show the results issued from the Holm procedure, and joins pairs of methods that do not yield statistically significant differences

these databases have been removed from the statistical tests), because after more than a month and a half of computation, we did not obtain any results from the code provided by the authors. However, we do not mean to imply that the EDSC method is computationally more expensive than our method or the others considered in the paper. Our method has been programmed to perform the cross validation process in parallel, whereas the code for EDSC has been used directly as provided by its authors. In this context, some experiments on the running times of each method have been performed and included as supplementary material in our website.[7] Note that these results are not conclusive and should be interpreted with care, since each method has been programmed in a different language and by a different programmer.

If we analyze the accuracy results, it can be seen that the ECDIRE method obtains the best results in 16 databases, only preceded by the RelClass method which obtains the best accuracy in 19 databases. The ECTS and the EDSC methods beat all the other methods in only 2 and 4 cases, respectively. Furthermore, it should be noted that, contrary to the other methods, in the ECDIRE method, it is common that at the end of the process some series remain unclassified, due to the unreliability of their class label (see values between parentheses in Table 1). In this work, we choose the worst possible validation scenario for our method and include unclassified examples as incorrectly classified instances. This includes the series that, at the end of the prediction process, do not pass the reliability tests but also includes those that are assigned with one of the initially discarded classes (see Step 1 of Sect. 4.1). However, it is possible that in some cases, the cost of making an error is larger than that of not providing a class value. Also, in some real contexts, outliers series that do not belong to any class could be present. In these cases, if a cost sensitive measure were applied, the results of the ECDIRE method regarding the committed error would further improve.

Figure 6 enables us to conclude that, most methods do not obtain significantly higher or lower results in terms of accuracy. The only method that, overall, obtains lower accuracy results than the other methods in this objective is EDSC.

If the results for earliness are observed in Table 2, it can be seen that ECDIRE obtains the lowest value in 22 databases out of 45, followed by EDSC, that obtains the best result in 17 cases. The other two methods obtain the best results in less cases, 6 (RelClass) and 1 (ECTS). If we observe the statistical tests for earliness, we can

[7] http://www.sc.ehu.es/ccwbayes/members/umori/ECDIRE/runtimes.

Table 3 Summary of domination counts for ECDIRE versus RelClas, ECTS and EDSC

	RelClass	ECTS	EDSC
ECDIRE	13/29/3	24/19/2	16/27/2

The first number corresponds to the number of times ECDIRE dominates the other method, the second number refers to the the cases in which the Pareto optimality criterion gives us no information (draws) and the third number counts the times the comparison method dominates ECDIRE

see that, our method is among the methods that obtain overall lower earliness values (ECDIRE and EDSC).

Since ECDIRE obtains good results in both objectives separately, we can conclude that it does not lean towards one of the objectives in particular. However, in order to compare the early classifiers and obtain conclusive results, we must take both objectives into account and consider them together by using domination counts. In Table 3 we provide a summary of the domination counts for the pairwise comparison between ECDIRE and the rest of the early classification proposals considered in this study. Based on the results shown in Table 3, it can be said that ECDIRE dominates the other methods much more often than it is dominated by them. Furthermore, we have also tested these results using the *Asymptotic nonrandomized UMP (ANU) test* proposed by (Putter 1955) obtaining significant differences in all cases using a significance value of α=0.05.[8] Hence, we conclude that our proposal improves the results obtained by the other state-of-the-art methods, and is thus a good solution for performing reliable and early classification of time series.

Additionally, if we calculate the times in which our method is dominated by any of the comparison methods (taking into account all the possible parameter configurations considered in the experimentation) we obtain a value of 8, which confirms the fact that our method is not dominated many times by the other methods, even if we allow the selection of different methods and parameter options for each database. If we calculate the same value for the rest of the methods, the lowest value that appears is 15, obtained by the RelClass method with $\tau = 0.1$ and the Naive Gaussian Quadratic set method. Moreover, if we calculate the times in which our method dominates all the remaining methods (considering all parameter configurations), we conclude that this happens in 5 databases. This value is larger than 0, which implies that our method dominates all the other methods with all their parameter configurations in at least one occasion. Additionally, this value is also larger than the number of times in which any other method considered dominates ECDIRE (see Table 3). All this indicates that our method is a useful early classifier. Moreover, if we calculate this value for all the rest of the methods, we obtain, at most, a value of 1, which is obtained in three cases: RelClass with $\tau = 0.5$ and with both the Gaussian Naive Bayes box method and the Naive Gaussian Quadratic set method, and the loose ECTS method with the support set to 0.

[8] The p values for these tests can be seen in http://www.sc.ehu.es/ccwbayes/members/umori/ECDIRE/pvalues.

Finally, we must include some information concerning the theoretical computational complexities of the methods. As previously mentioned, the running times of the training phases of the early classifiers are not crucial in this context, as the models are trained in batch mode. However, what should be taken into account is the time that each method needs to provide an answer (class value or abstention) at a given time t. This is because the data arrives in a streaming manner and the methods should be able to react before the next data point arrives. In this context, we have calculated the theoretical worst case complexities of this part of the execution for each of the considered early classifiers, obtaining the following values: O($\max\{n \cdot L, n^2\}$) for ECDIRE, O($n \cdot L$) for ECTS, O($S \cdot l$) for EDSC, O($\max\{n \cdot D, D^2\}$) for RelClass using dimensionality reduction and O($\max\{n \cdot L, L^2\}$) for RelClass without dimensionality reduction. n is the number of series in the training set, L is the length of the series in the training set (we suppose for the sake of simplicity that all series are of the same length), S is the number of shapelets that the EDSC method includes into its library, l is the length of the largest shapelet in this library, and D is the dimension of the time series after LDG dimensionality reduction in RelClass. Note that the complexities do not differ much from method to method. Some experiments that show the actual runtimes for this part of the prediction phase can be found in our website.[9]

7 Case study: early classification of bird songs

In addition to the experiments shown for the UCR archive, and taking into account the ubiquity of streaming data, we consider it imperative to show how our system could be applied to this type of data, which is of unknown and possibly infinite length. As a case study, we have used a small size dataset that does not allow forceful statistically significant claims. Our intention is simply to show the utility of our ideas outside of a lab setting.

In various parts of the world, automatic audio and video monitoring stations are set up in an attempt to record targeted species. For example, in Myanmar (Burma), the Myanmar Jerdon's babbler, *Chrysomma altirostre* (Collar et al. 2001), was believed to be extinct, but in 2015 a small population was discovered (Dell'Amore 2015). Discovery of such tiny populations is often made with a targeted search, placing ruggedized video monitoring stations in the field. Since such devices have limited computational resources, especially memory and battery-life, they are often designed to be triggered by sound. A simple volume threshold is not suitable because it is easily tricked by wind noise or non-target species. Clearly, it would be better for the video recording to be triggered by the target species call. This appears to offer a paradox, how can one get a recording of a bird that may be extinct? In many cases, it is possible to get the recoding of a closely related species. For example, in the above case, there is a closely related species, the Sind Jerdon's babbler (*Chrysomma altirostre scindicum*) that is common in the Indus basin of Pakistan.

[9] http://www.sc.ehu.es/ccwbayes/members/umori/ECDIRE/runtimes.

Given this, we can approximately learn the typical call of the target species, and use it to trigger image/video recording. This is a natural domain for early classification; we would like to begin recording as soon as possible, before the bird can flit away.

Note that this problem setting differs slightly from the common early classification problem. In this case, we have a unique stream of data of unknown and possibly infinite length, and the aim is to detect certain patterns of interest within the stream as early as possible.

This problem bears some similarities with the problem of stream classification. Stream classification is a supervised learning task where the objective is to learn a model that will classify each data point from an incoming (possibly infinite) stream of data (Gaber et al. 2007). However, in many cases, such as in online human activity recognition, one single data point from the stream is not informative enough to provide a class value (Lara and Labrador 2012). In these cases, the stream is divided into a set of time windows and the data enclosed within each window is considered an instance. The size of these windows determines how early we will be able to make the classification. However, this parameter is typically chosen using domain knowledge or by directly analyzing the performance of different window sizes (Lara and Labrador 2012). In this context, our method can be used to automatically fix the window size in order to provide the class predictions as early as possible while maintaining a suitable level of accuracy.

Most early classification methods in the literature can not deal with this specific problem setting but, the design of the ECDIRE framework provides a direct extension to this very common problem setting, unlike most early classification methods proposed in the literature.

To demonstrate this, we performed the following experiment. We select two different birds from the same species, namely the White Crowned Sparrow species (*Zonotrichia leucophrys pugetensis*) from the Xeno-Canto database (Xeno-canto Foundation 2005). Although the calls of birds from the same species are similar to each other, there are clear differences between the calls of each individual bird. As such, provided a stream containing forest noise and some occasional bird calls, the goal is to try to detect and identify each bird call as early in time as possible.

We exploit the fact that we can convert high-dimensional (44,100 Hz) audio into a low dimensional Mel-frequency cepstral coefficients (MFCC) space (100 Hz). It has long been known that the MFCC space is amenable to bird song classification (Kogan and Margoliash 1998; Ulanova et al. 2015). While most classification algorithms attempt to use up to twelve coefficients, for simplicity and visual intuitiveness we consider only one here. As Fig. 8 shows, a single MFCC coefficient does suggest that little interclass variability is present, which bodes well for the task at hand.

To apply the ECDIRE method, we build a training set consisting of 5 and 8 calls from each of the birds of length 200, respectively. As shown in Sect. 4.1, we apply a cross validation process to this data and build the corresponding timeline. From this process, we obtain one safe timestamp for each class, with values of 15 % (t_1) and 20 % (t_2) of the length of the bird calls, respectively. Additionally, we calculate the reliability thresholds as explained in Sect. 4.1 for these early timestamps.

Now, we use the information obtained from this process to obtain class predictions of a new testing stream of data. In this experiment, the length of the stream is of 1600

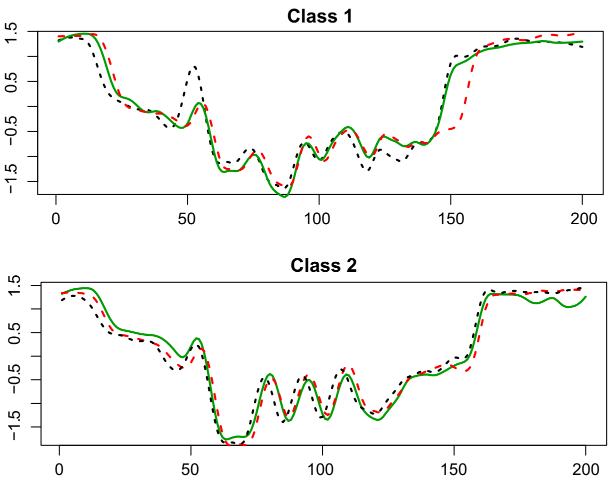

Fig. 8 Three example calls from each class

points, but the method could be applied identically to a stream of unknown or even infinite length. As previously commented, this stream will contain a few bird calls and background noise from the forest in between the calls. Although the ECDIRE method is initially defined to classify finite time series, the set of "safe" timestamps for each class provides a direct way to apply it to this other problem setting. This is because the "safe" timestamps directly define suitable sizes for the sliding windows that will be used to analyze the incoming stream.

Thus, to identify the call of the first bird, we analyze the stream with a sliding window of size t_1 (15 % of the length of the pattern, thus 30 data points). Every time a new data point arrives, we slide the window one position, and, by using a GP classifier built with all the training data for the corresponding earliness value t_1 (see Sect. 4.1), we obtain a class prediction for the data section enclosed within. Note that since the test stream is 1600 data points long and the sliding window is 30 points wide, we will perform 1571 class predictions. To finish, as shown in Sect. 4.2, we apply the reliability test and decide whether we make a final prediction or choose to abstain. In this case, given the high level of noise present in the incoming testing stream, instead of using the minimum to calculate the reliability thresholds, we decide to use a stricter threshold and calculate the mean of the differences between the posterior probabilities of the target class and the next most probable class obtained in the training phase. In Fig. 9 we show the results obtained for the first bird.

It can be seen in Fig. 9 that the first bird is identified perfectly and very early in time, with a window size of only 15 % of the length of the pattern. Additionally, for this bird, our method issues no false positives.

For the second bird, we follow the same procedure, but take the sliding window of size t_2 (20 % of the length of the patterns, thus 40 data points), provided by the ECDIRE method. Note that with this setting, in this case, we make 1561 class predictions. By using the mean difference between the posterior probabilities of the target class and the

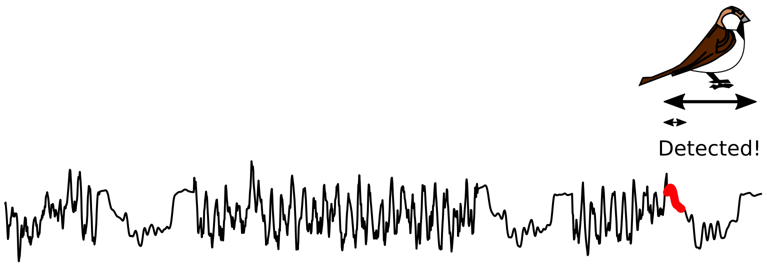

Fig. 9 Detection and identification of the first bird by using the ECDIRE method, using the mean to calculate the reliability threshold. The presence of this bird in the stream is represented by an image of a bird, and the first window in which the bird is identified is shown by a *bold line*. The *arrows* indicate the length of the call and the portion used to detect it, respectively (from *top* to *bottom*)

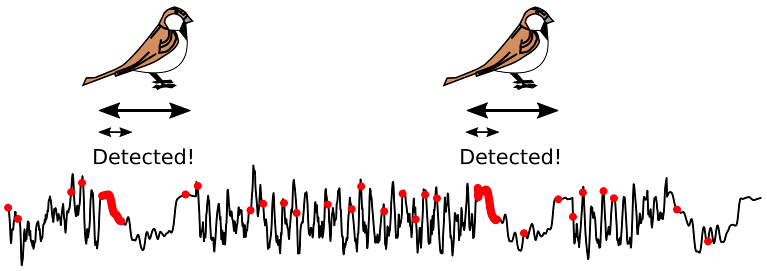

Fig. 10 Detection and identification of the second bird by using the ECDIRE method, using the mean to calculate the reliability threshold. The presence of this bird in the stream is represented by an image of a bird, and the first window in which the bird is identified is shown by a *bold line*. The *arrows* indicate the length of the call and the portion used to detect it, respectively (from *top* to *bottom*). False positives are represented by round points in the stream. Note than when the method issues a false positive, it usually commits the same mistake in several consecutive windows. In the image, for the sake of clarity, only the first window is *marked*

next probable class obtained in the training phase to calculate the reliability threshold, we obtain the result shown in Fig. 10.

This bird is also identified correctly in all cases, with a window size of only 20 % of the length of the bird call. However, contrary to the first case, for this bird we obtain many false positives (176 from 1561 predictions). The reason for this is that, when the training set is very small and unbalanced, the GPs that we use within ECDIRE tend towards the largest and more variable classes, in this case that which corresponds to the second bird. This results in many false positives for this bird and none for the other, especially if the reliability thresholds are not very high.

The results obtained are still better than those that would be obtained by a simple volume threshold, which would not be able to distinguish between birds and would constantly be triggered by the background forest noise, but they could be improved by better tuning the reliability threshold. For example, if we take the median instead of the mean, we obtain a stricter reliability threshold and, thus, the results shown in Fig. 11. In this case, the number of false positives is much lower (7 false positives from 1561 predictions) but the method fails to identify the bird on one of its appearances.

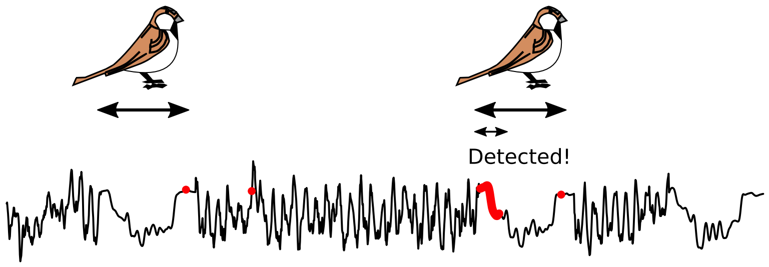

Fig. 11 Detection and identification of the second bird by using the ECDIRE method, using the median to calculate the reliability threshold. The presence of this bird in the stream is represented by an image of a bird, and the first window in which the bird is identified is shown by a *bold line*. The *arrows* indicate the length of the call and the portion used to detect it, respectively (from *top* to *bottom*). False positives are represented by round points in the stream. Note than when the method issues a false positive, it usually commits the same mistake in several consecutive windows. In the image, for the sake of clarity, only the first window is *marked*

We insist on the fact that the aim of this section is to simply demonstrate the utility and direct application of ECDIRE to a problem of this type, and that the small size of the training set does not allow us to make any claims regarding the results. Specifically, recall that the reliability thresholds are calculated based on the differences between the posterior probabilities obtained in the training process. The extremely small number of training instances make the mean and median values not very robust in this example, which is why the results change so drastically. However, with a larger training set, we would be able to better tune the reliability thresholds in order to remove the false positives while retaining the correct class identifications. Optimally tuning the reliability threshold based on the level and type of noise in the incoming stream could be an interesting area of study, which is proposed as future work.

8 Conclusions and future work

In this paper, we have proposed a method for early classification of time series based on probabilistic classifiers. Our proposal incorporates two strategies that deal with two of the main issues that arise with other early classification methods published in the literature. On the one hand, in order to avoid excessive calculations in the prediction phase, the discriminativeness of the classes is analyzed at different time intervals and a methodology is proposed to identify the time instant at which we can begin to make accurate predictions for each class. This procedure allows a direct application to infinite streams of data, which had not been dealt with in other early classification works. On the other hand, the goodness of the issued class labels is controlled, discarding predictions that do not yield a sufficiently large class probability.

Contrary to the other early classification methods used for comparison, the proposal fulfills the three desirable characteristics for early classification methods recommended in Ghalwash et al. (2014). First, the method obtains competitive results with respect to the earliness of the prediction in comparison with other state-of-the-art early classification methods. Indeed, the few methods that obtain similar results in earliness are weaker than ECDIRE with respect to accuracy. When evaluating the methods from a multi-objective point of view, it can be concluded that the ECDIRE method dominates the rest of the methods much more frequently than it is dominated. All this proves the usefulness of the method to obtain reliable and early classifications of time series.

Secondly, with regards to understandability, our method is able to discover at which time instant each class can be reliably discriminated from other classes. This information is valuable and may help the user to understand the underlying patterns in the data. In this sense, it could be interesting to perform a theoretical analysis of the relation between the characteristics of the database and the evolution of the accuracy over time, which could lead to characterizing the level of conflict between the two objectives and could provide further information to the users.

Additionally, the method can be applied by simply setting the $perc\_acc_c$ parameter and a grid of early timestamps to consider. The first parameter can easily be understood and fixed by any user, expert or otherwise. The granularity can be set to 5 %, which has provided good results in the experimentation, but could also be defined based on domain knowledge or depending on other requirements. More experienced users may also modify other parameters such as the distance measure d, the kernel function, etc. but this is not a requirement. In this line, an obvious proposal for future work consists of studying the behavior of different distance measures and kernels within this framework. Additionally, it could also be interesting to try to find the most suitable granularity for a certain database.

As a last point, a straightforward measure for uncertainty of the predictions issued by our framework is provided by the probabilities extracted from the classifiers employed within the framework. A more informative uncertainty measure could incorporate additional knowledge acquired in the learning phase, such as information about the discriminativeness of the classes in the database in different time intervals. The design of this measure is proposed for future elaboration.

Acknowledgements We are deeply grateful to Jerónimo Hernández-González for his helpful comments. Also, thanks to Nurjahan Begum and Liudmila Ulanova for their useful advice, and for formatting and preparing the data used in the bird call identification case study. We would also like to thank the UCR archive and the Xeno-Canto Foundation for providing access to the data used in this study. This work has been partially supported by the Saiotek and IT-609-13 programs (Basque Government), TIN2013-41272P (Spanish Ministry of Science and Innovation) and by the NICaiA Project PIRSES-GA-2009-247619 (European Commission). Usue Mori holds a grant from the University of the Basque Country UPV/EHU.

References

Bregón A, Simón MA, Rodríguez JJ, Alonso C, Pulido B, Moro I (2006) Early fault classification in dynamic systems using case-based reasoning. In: CAEPIA'05-Proceedings of the 11th Spanish association conference on current topics in artificial intelligence. pp 211–220

Calvo B, Santafé G (2015) scmamp: Statistical comparison of multiple algorithms in multiple problems. R package version 0.2.2. https://github.com/b0rxa/scmamp

Collar NJ (2001) Chrysomma altirostre. In: Collar NJ, Andreev A, Chan S, Subramanya S, Tobias J, Tobias J (eds) Threatened birds of Asia: the birdlife international red data book. BirdLife International, Cambridge, pp 2112–2119

Dell'Amore C (2015) 'Extinct' bird rediscovered in Myanmar, surprising scientists. http://news.nationalgeographic.com/news/2015/03/150305-birds-extinct-rediscovered-myanmar-burma-animals-science/

Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30

Evans RS, Kuttler KG, Simpson KJ, Howe S, Crossno PF, Johnson KV, Schreiner MN, Lloyd JF, Tettelbach WH, Keddington RK, Tanner A, Wilde C, Clemmer TP (2015) Automated detection of physiologic deterioration in hospitalized patients. J Am Med Inform Assoc 22(2):350–60. http://www.ncbi.nlm.nih.gov/pubmed/25164256

Gaber MM, Zaslavsky A, Shonali K (2007) A survey of classification methods in data streams. In: Data streams. Vol. 31. pp 39–59. http://link.springer.com/chapter/10.1007/978-0-387-47534-9_3

Ghalwash MF, Radosavljevic V, Obradovic Z (2014) Utilizing temporal patterns for estimating uncertainty in interpretable early decision making. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining—KDD '14. ACM Press, New York, pp 402–411

Ghalwash MF, Ramljak D, Obradovic Z (2012) Early classification of multivariate time series using a hybrid HMM/SVM model. In: IEEE international conference on bioinformatics and biomedicine. pp 1–6

Girolami M, Rogers S (2006) Variational Bayesian multinomial probit regression with Gaussian process priors. Neural Comput 18:1790–1817

Graepel T, Herbrich R, Bollmann-sdorra P, Obermayert K (1998) Classification on pairwise proximity data. NIPS. The MIT Press, Cambridge, pp 438–444

Hatami N, Chira C (2013) Classifiers with a reject option for early time-series classification. In: IEEE symposium on computational intelligence and ensemble learning (CIEL). pp 9–16

He G, Duan Y, Peng R, Jing X, Qian T, Wang L (2015) Early classification on multivariate time series. Neurocomputing 149:777–787

Kadous MW, Sammut C (2005) Classification of multivariate time series and structured data using constructive induction. Mach Learn 58(2–3):179–216

Kate RJ (2015) Using dynamic time warping distances as features for improved time series classification. Data Mining and Knowledge Discovery. http://link.springer.com/10.1007/s10618-015-0418-x

Keogh E, Zhu Q, Hu B, Y., H., Xi X, Wei L, Ratanamahatana CA (2011) The UCR time series classification/clustering homepage. www.cs.ucr.edu/~eamonn/time_series_data/

Kogan Ja, Margoliash D (1998) Automated recognition of bird song elements from continuous recordings using dynamic time warping and hidden Markov models: a comparative study. J Acoust Soc Am 103(4):2185–2196

Lama N, Girolami M (2014) vbmp: variational Bayesian multinomial probit regression. R package version 1.34.0. http://bioinformatics.oxfordjournals.org/cgi/content/short/btm535v1

Lara OD, Labrador MA (2012) A survey on human activity recognition using wearable sensors. IEEE communications surveys & tutorials, pp 1–18. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6365160

Li C, Khan L, Prabhakaran B (2006) Feature selection for classification of variable length multiattribute motions. Knowl Inf Syst 10(2):163–183

Parrish N, Anderson HS, Hsiao DY (2013) Classifying with confidence from incomplete information. J Mach Learn Res 14:3561–3589

Pree H, Herwig B, Gruber T, Sick B, David K, Lukowicz P (2014) On general purpose time series similarity measures and their use as kernel functions in support vector machines. Inf Sci 281:478–495

Putter J (1955) The treatment of ties in some nonparametric tests. Ann Math Stat 26(3):368–386

Rasmussen CE, Williams CKI (2006) Gaussian processes for machine learning. The MIT Press, Cambridge. www.GaussianProcess.org/gpml

Rodríguez JD, Pérez A, Lozano JA (2013) A general framework for the statistical analysis of the sources of variance for classification error estimators. Pattern Recognit 46(3):855–864

Stathopoulos V, Zamora-Gutierrez V, Jones KE, Girolami M (2014) Bat call identification with Gaussian process multinomial probit regression and a dynamic time warping kernel. In: Proceedings of the 17th international conference on artificial intelligence and statistics. Vol. 33, pp 913–921

Ulanova L, Begum N, Keogh E (2015) Scalable clustering of time series with U-shapelets. In: SIAM international conference on data mining (SDM 2015)

Wang X, Mueen A, Ding H, Trajcevski G, Scheuermann P, Keogh E (2012) Experimental comparison of representation methods and distance measures for time series data. Data Min Knowl Discov 26(2):275–309

Xeno-canto Foundation (2005) xeno-canto: Compartiendo cantos de aves de todo el mundo. http://www.xeno-canto.org/

Xing Z, Pei J, Keogh E (2010) A brief survey on sequence classification. ACM SIGKDD Explor Newsl 12(1):40

Xing Z, Pei J, Yu PS (2011a) Early classification on time series. Knowl Inf Syst 31(1):105–127

Xing Z, Yu PS, Wang K (2011b) Extracting interpretable features for early classification on time series. In: Proceedings of the eleventh SIAM international conference on data mining. pp 247–258

Ye L, Keogh E (2009) Time series shapelets : a new primitive for data mining. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining. pp 947–956