Information Processing and Management 60 (2023) 103465

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/ipm

CALIMERA: A new early time series classification method

Jakub Michał Bilski, Agnieszka Jastrzębska *

Faculty of Mathematics and Information Science, Warsaw University of Technology, ul. Koszykowa 75, 00-662 Warsaw, Poland

ARTICLE INFO

Keywords:
Early time series classification
Calibration
Decision-making
Cost function

ABSTRACT

Early time series classification is a variant of the time series classification task, in which a label must be assigned to the incoming time series as quickly as possible without necessarily screening through the whole sequence. It needs to be realized on the algorithmic level by fusing a decision-making method that detects the right moment to stop and a classifier that assigns a class label. The contribution addressed in this paper is twofold. Firstly, we present a new method for finding the best moment to perform an action (terminate/continue). Secondly, we propose a new learning scheme using classifier calibration to estimate classification accuracy. The new approach, called CALIMERA, is formalized as a cost minimization problem. Using two benchmark methodologies for early time series classification, we have shown that the proposed model achieves better results than the current state-of-the-art. Two most serious competitors of CALIMERA are ECONOMY and TEASER. The empirical comparison showed that the new method achieved a higher accuracy than TEASER for 35 out of 45 datasets and it outperformed ECONOMY in 20 out of 34 datasets.

1. Introduction

Early time series classification (ETSC) is a variant of the classification process in which an algorithm is equipped with a decision-making mechanism that allows it to halt screening a given time series at the point when there is an acceptable level of probability that the proposed class label is correct. There exist many different ways to measure the performance of ETSC, but the general agreement is that making a method stop faster is contradictory in nature to making classification more accurate (Yao, Chen, & Yao, 2021); the fewer data the method has seen, the less accurate (but faster) classification will be. Applications of ETSC include domains where making a fast decision can save resources or significantly change the outcome of the process, such as gas leakage detection (Hatami & Chira, 2013), early diagnosis (Tong et al., 2022), earthquakes prediction (Fauvel et al., 2020), human gesture recognition (Gupta et al., 2022), and electricity demand prediction (Gu et al., 2021).

Most of the existing ETSC methods (Achenchabe, Bondu, Cornuéjols & Dachraoui, 2021; Hartvigsen, Sen, Kong, & Rundensteiner, 2019; Huang, Yen, & Tseng, 2022a, 2022b; Karim, Majumdar, Darabi, & Harford, 2019; Lv, Hu, Li, & Li, 2019; Ren, Wang, & Zhao, 2022; Salman, Elhajj, Chehab, & Kayssi, 2022; Schäfer & Leser, 2020) are composed of two steps. The first is responsible for generating features from the currently available time series. The second is concerned with stopping based on those features. The efficiency of cooperation of these two steps is purely heuristic – the first step is not aware that its results will be used for the stopping decision, while the second step does not control how the series are processed. In fact, the second step merely stops some process at such a moment that it generates the lowest cost function (or other chosen metric). This division, however effective, indicates a severe weakness of the aftermentioned methods and a lack of benefit from focusing on ETSC. One can imagine a real ETSC method that dynamically chooses the mode of time series processing to maximize the stopping module's efficiency. To our knowledge, no such method exists yet.

[*] Corresponding author.
E-mail address: A.Jastrzebska@mini.pw.edu.pl (A. Jastrzębska).

https://doi.org/10.1016/j.ipm.2023.103465
Received 20 February 2023; Received in revised form 3 July 2023; Accepted 11 July 2023
Available online 24 July 2023

We take a step in a different direction. We propose to formalize the stopping decision as a separate problem, independent from time series classification. Based on this assumption, a new algorithm is delivered. Experiments show that this new "agnostic" approach achieves results better than the state-of-the-art. This shows that the dedicated ETSC solutions were not able to effectively use the knowledge about what is being stopped. Thus, considering early stopping in a more general setting becomes an attractive prospect for ETSC methods that follow the two-step pattern. It opens the way for many new applications and allows us to compare ETSC methods more easily.

The new method introduced in this paper was named CALIMERA, short for Calibrated eArLy tIMe sERies clAsifier.[1] The stopping condition in CALIMERA was designed by generalizing the problem of ETSC into a problem of minimizing a cost function by performing an action at the best time. We use the mechanism of classifier calibration to acquire information about the expected classification performance, which then is the base for making the waiting/stopping decision. We create a complete ETSC pipeline by utilizing one of the best-performing methods for time series classification, MINIROCKET (Dempster, Schmidt, & Webb, 2021).

To the best of our knowledge, this is the first study in which classifier calibration is used for ETSC to estimate probabilities of making an error. We show that this technique allows us to learn the stopping condition better and can be easily incorporated into the learning scheme of a new classification algorithm we propose.

Essential advantages of CALIMERA include:

- the ability to perform both binary and multiclass classification (which is not always the case for ETSC algorithms);
- a clear separation of the stopping/waiting decision step and the classification step;
- *informed* stopping/waiting decision step, in which each decision is made based on estimated classifier's accuracy produced by a calibrator;
- involvement of a customizable cost function with which a user can specify whether accuracy or earliness is more important;

Our code was made publicly available at bit.ly/3I1HouE. In an additional effort aimed at supporting the replicability of our results, CALIMERA was added to a publicly available experimental benchmark (Kladis, Akasiadis, Michelioudakis, Alevizos, & Artikis, 2021).

The structure of this paper is as follows. Section 2 recalls key literature positions relevant for this study. Section 3 introduces background knowledge concerning the development of CALIMERA. Section 4 presents the new model. Section 5 compares the new solution to the current state-of-the-art. Section 6 concludes the paper.

2. Literature review

As Gupta, Gupta, Biswas, and Dutta (2020) and Lv et al. (2019) claim, Xing, Pei, and Yu (2009) were the first to propose a solution to ETSC by establishing a trade-off between accuracy and earliness. Their solution, called ECTS (short for Early Classification on Time Series), used a clustering method and established a measure named Minimum Prediction Length (MPL), meaning the minimal subsequence of time series that can represent the whole time series without compromising accuracy. The assumed goal was to keep accuracy on the same level while minimizing earliness. However, as the model was not ideal (learned on a training data), it could never guarantee a perfect retention of accuracy. Another method, Early Distinctive Shapelet Classification (Xing, Pei, Yu, & Wang, 2011) (EDSC), used shapelets to create a set of patterns characteristic of samples from a given class. A specific distance between the shapelets and the sequence allowed to estimate probability of accurate classification. ECTS and EDSC did not propose a clear definition of ETSC as an optimization goal, but merely provided some trade-offs between accuracy and earliness that were results of their mode of operation.

Lv et al. (2019) proposed Effective Confidence-based Early Classification (ECEC), which estimated confusion matrices for all possible stopping points, then searched for the confidence threshold that minimized a cost function representing the desirable trade-off between accuracy and earliness. Thus, ETSC could finally be properly defined, as a task of minimizing said function. The same cost function was later used in TSOCF (Zhang & Wan, 2022), which selected lengths of discriminatory shapelets in the stopping phase to minimize it. Another method designed based on a clear definition of ETSC was TEASER (Schäfer & Leser, 2020), which used classifiers to generate scores for each class, which were then passed to master classifiers that learned at what conditions the scores should be trusted. TEASER was designed to maximize the harmonic mean of accuracy and earliness. However, perhaps due to the lack of additivity, this metric was used only when adjusting a single parameter.

Some of the ETSC solutions used neural networks to classify or decide on stopping. One of them was Multivariate Long Short Term Memory Fully Convolutional Network (MLSTM) (Karim et al., 2019). MLSTM was initially introduced as an algorithm for Multivariate Time Series Classification and later adapted to ETSC in the experimental framework by Kladis et al. (2021) by selecting the prefix length that produced the best harmonic mean of accuracy and earliness. A serious limitation of this approach was that all sequences were stopped at the same moment. MLSTM used a concatenation of convolutional layers followed by a Squeeze-and-Excite block (Hu, Shen, & Sun, 2018) and an LSTM network (Van Houdt, Mosquera, & Nápoles, 2020/12/01).

Another solution to ETSC that used LSTM was EARLIEST (Hartvigsen et al., 2019). EARLIEST classified samples with LSTM, then used its hidden states to decide on stopping. Unfortunately, authors of EARLIEST provided limited experimental evidence of the model's quality, and a recent automated framework for selecting hyperparameters, MultiECTS (Ottervanger, Baratchi, & Hoos, 2021) revealed that EARLIEST usually performs substantially worse than its competitors.

[1] CALIMERA pronounced in English sounds like $\kappa\alpha\lambda\eta\mu\acute{\varepsilon}\rho\alpha$, which is Greek *good morning*.

To better support real applications, Dachraoui, Bondu, and Cornuéjols (2015) proposed a customizable cost function for ETSC, which was later chosen as a minimization criterion in ECONOMY (Achenchabe, Bondu, Cornuéjols & Dachraoui, 2021). ECONOMY used TSFEL (Barandas et al., 2020) and XGBoost (Su et al., 2023), followed by a custom stopping policy, which limited the method to binary classification. A different approach to support the end-user was chosen by authors of MultiETSC (Ottervanger et al., 2021), who created an automated machine learning framework to generate a broad spectrum of solutions representing different accuracy vs earliness trade-offs. This approach implicitly assumed that the cost of delaying a decision was linear in time, a fact pointed out in Achenchabe, Bondu, Cornuéjols and Dachraoui (2021). In our opinion, the evaluation criterion used in MultiETSC was controversial – creating a Pareto set based on a test data meant ignoring bad solutions, effectively promoting methods that generated results of high variance.

Some authors considered variants of ETSC, like the Early and Revocable Time Series Classification (Achenchabe, Bondu, Cornuejols, & Lemaire, 2022), where a once made classification decision could be revoked for an additional cost. Another one was early classification of irregular time series, introduced by Hartvigsen et al. in Hartvigsen, Gerych, Thadajarassiri, Kong, and Rundensteiner (2022), where they presented Stop&Hop, an algorithm designed for that setting. A number of ETSC solutions were created to support a specific application, leveraging domain knowledge to outperform general-purpose algorithms. Reinforced Siamese network with Domain knowledge regularization (Ren et al., 2022) utilized Siamese networks and domain knowledge about feature distribution to create a method for Early Diagnosis. Snippet Policy Network (Huang et al., 2022b) and Constraint-based Knee-guided Neuroevolutionary Algorithm (Huang et al., 2022a), both designed by Huang et al. diagnosed cardiovascular diseases based on electrocardiogram data. Salman et al. (2022) considered the problem of internet traffic early classification, while End-to-End Learned Early Classification of Time Series (Rußwurm et al., 2023) performed early in-season crop type mapping.

In our method, we utilize the technique of calibration. Calibration of a classifier means turning scores into class membership probabilities. A well-calibrated classifier outputs predicted probabilities distributed approximately the same as true class labels. One of the earlier binary classifier calibration methods was logistic calibration, introduced by Platt (2000). Another early method of multiclass calibration was based on the Dirichlet Calibration, introduced in Gebel (2009), which assumes that the predicted probability vectors for each class have a Dirichlet distribution. Later, Kull, Filho, and Flach (2017) developed a technique that produced calibration maps based on the beta distribution. This approach was empirically proven to be superior to logistic calibration. Unfortunately, it was suitable only for binary classifiers. Recently, Böken (2021) showed that Platt scaling (also called sigmoid or logistic calibration) allows for achieving equivalent results as beta calibration but is suitable for multiclass problems. The newest advances in classifier calibration include studies on the application of classifier calibration to datasets which contain instances challenging to classify (Paiva, Moreno, Smith-Miles, Valeriano, & Lorena, 2022) or as a method for instance space analysis (Muñoz et al., 2021). Furthermore, extensions of classifier calibration were proposed, for example, constructed with supplementary cost functions (Wang et al., 2022).

3. Background knowledge

This section discusses notions and algorithms that constitute the backbone of the proposed approach. In particular, Section 3.1 presents a cost function used in ETSC and the measure of earliness. Section 3.2 addresses the theoretical motivation behind the separation of the stopping decision from the classification step in ETSC.

3.1. Cost function

Behind each ETSC problem, there is an underlying decision process that the classification decision is a part of. It is the characteristics of this process that allow for some interchangeability between accuracy and *earliness* of the prediction, thus, making it possible to create a single optimization goal to balance the two.

Earliness of ETSC is defined as

$$earliness = \frac{1}{|Y|} \sum_{y_i \in Y} \frac{\tau_i}{T_i}, \tag{1}$$

where Y is the set of labels, τ_i is the last timestamp that was used by the method during classification of the ith sample, and T_i is the maximal timestamp in the sample.

Harmonic mean of accuracy and earliness HM is defined as

$$HM = \frac{2 \cdot (1 - earliness) \cdot accuracy}{(1 - earliness) + accuracy}. \tag{2}$$

Harmonic mean of accuracy and earliness is widely used in literature (Kladis et al., 2021; Ottervanger et al., 2021; Schäfer & Leser, 2020) whenever there is a need to measure quality of an ETSC solution with a single metric. Please note that harmonic mean does not allow for parametrization, which makes it less useful in a practical setting. Another serious weakness is its lack of additivity, which makes it impossible to translate it into a local optimization criterion.

General ETSC Cost Function is defined as

$$C_G = \frac{1}{|Y|} \sum_{y_i \in Y} C_d(\tau_i) + C_m(\bar{y}_i | y_i), \tag{3}$$

where Y is the set of labels, τ_i is the last timestamp that was used by the method during the classification of the i-th sample, and $\bar{y}_i$ is the label assigned to the i-th sample by the method. $C_m(\bar{y}|y) : Y \times Y \to R$ is the *misclassification cost function* that defines the cost of selecting label $\bar{y}$ when the correct label is y. $C_d(t) : R \to R$ is the *delay cost function*. C_d is non decreasing over time.

Using General ETSC Cost Function is more intuitive than using harmonic mean, as it requires only predefined penalties for classification errors and time delay, both of which can be measured in a local currency. The absence of accuracy and earliness in the cost definition is an advantage, as those concepts would first have to be explained to the user. We argue that because of its wide possibilities of parametrization and additivity, this cost function is better-suited to be the optimization goal for a new ETSC method.

However, the versatility of General ETSC Cost Function does come at a price. A significant part of it is tied to the ability of setting separate costs for different label mismatches. While defining the C_m matrix can be effectively done with domain knowledge about similarities between classes (Mienye & Sun, 2021), minimizing the resulting cost function may be challenging in terms of implementation. Because of that, we follow (Achenchabe, Bondu, Cornuéjols & Dachraoui, 2021) in limiting the misclassification cost matrix to be all ones apart from zeros on the main diagonal. We do not make additional assumptions about the delay cost function.

3.2. Rephrasing the stopping decision in ETSC as a separate problem

Let us consider a supervised problem of choosing the best moment to perform an action in an online setting. Given a vector of features and the current timestamp, the task is to decide whether to perform an action or wait until the next timestamp. When the final timestamp T is reached, an action must be performed, which ends the process and generates a cost. The mode of cost generation can be learned from training data. Each sample in the training data is a vector of length T. Each vector element contains two elements: a vector of features and a cost of performing an action. The goal is to minimize the total cost on a test dataset.

Please note that the delay cost is not mentioned in this problem statement. If waiting should generate a cost, it can be included in the cost of performing an action. In the case of the method presented in this paper, this is equivalent to fitting an intercept in a cost prediction module at timestamp t.

In the existing methods, the choice of the best timestamp for classification was usually treated as a task separate from classification; see Achenchabe, Bondu, Cornuéjols and Dachraoui (2021), Lv et al. (2019), Schäfer and Leser (2020), Xing et al. (2009). However, none of these papers considered the problem in complete isolation. Doing that not only makes the presented approach clearer, but it also allows to use the stopping method in a different setting; it can be applied to any problem where an optimal moment to take some one-time action must be found. Some intuitive examples of tasks include waiting for the right time to substitute a player during a football match, replacing old hardware in a company before it starts breaking down, or launching an anti-aircraft missile to have the highest chance of hitting the target.

There is one crucial limitation to the problem statement described in this paragraph; it implicitly assumes that the total cost function is a sum of costs generated for all samples. This means that the cost function has to be additive with regard to individual samples. This condition is satisfied by the General ETSC Cost Function (Eq. (3)) that the new method is concerned with. However, it is not possible to maximize the harmonic mean of accuracy and earliness (Eq. (2)) using this framework. In our opinion, the lack of additivity in the latter metric is a fatal flaw that makes it useful only for very heuristic methods like TEASER.

4. Method

This section presents CALIMERA, the new algorithm for ETSC. Section 4.1 gives an overview of the model architecture. In subsequent sections, we discuss the building blocks of this architecture. Section 4.2 outlines how we use classification scores to produce features which are a prerequisite for early stopping decision computation. Section 4.3 addresses classifier learning strategy. Section 4.4 presents the notion of *cost difference*, which is the backbone of the early stopping mechanism. Section 4.5 presents how to compute the cost function. Section 4.6 shows how to employ the cost function to execute the early stopping in ETSC.

4.1. Model architecture

The method proposed in this paper approaches the problem of ETSC by first transforming the data into an input that fits the definition presented in Section 3.2. Then, the problem is solved by a stopping module. These two steps are distinctly separated, as visualized on Figs. 1 and 2 . The moment a decision to stop the process is made, a label produced by the classifier is outputted as the final result. The method can be summarized as a list of instructions:

1. Take the currently available parts of samples X_t, where t is the current timestamp.
2. Calculate features V_t for the stopping decision.
3. Consult the stopping module with the features V_t. For the samples where it decided to wait and t is lower than the maximal timestamp T, wait until the next timestamp and proceed to point 1.
4. Perform classification on X_t and return the predicted labels.

Algorithm 1 contains a detailed description of the algorithm in the form of a pseudo-code. The model architecture is additionally visualized in Fig. 1 and 2.

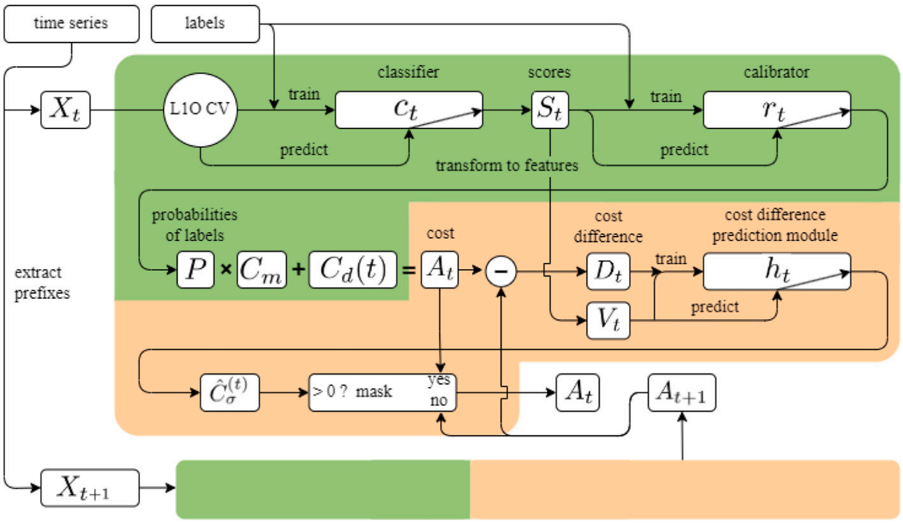

Fig. 1. CALIMERA – training procedure. The part specific to classification is marked in green, while the part responsible for stopping is marked in orange. c_t = classifier (MINIROCKET), r_t = calibrator (logistic regression), h_t = cost difference prediction module (kernel ridge regression), C_m = cost of misclassification, $C_d(t)$ = cost of delay, L1OCV = leave-one-out cross-validation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

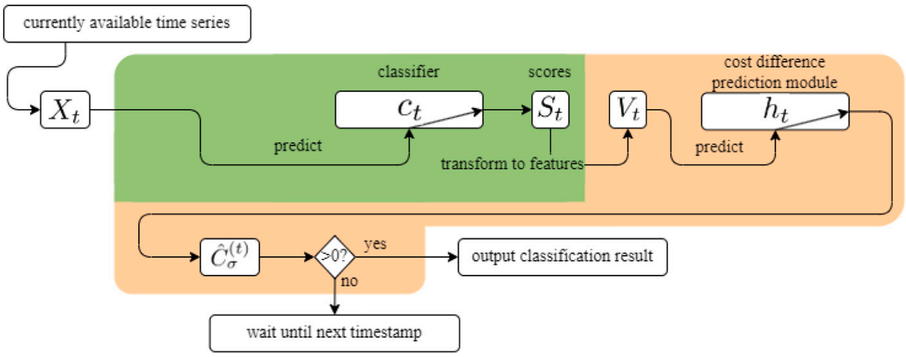
currently available time series
X_t
predict
classifier
c_t
scores
S_t
transform to features
V_t
cost difference prediction module
h_t
predict
$\hat{C}_\sigma^{(t)}$
>0?
yes
no
output classification result
wait until next timestamp

Fig. 2. CALIMERA – model architecture. The part specific to classification is marked in green, while the part responsible for stopping is marked in orange. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.2. Generating features for cost difference prediction

In many existing early time series classifiers, the main base for the stopping/waiting decision are scores returned by a classifier (Achenchabe, Bondu, Cornuéjols & Dachraoui, 2021; Hartvigsen et al., 2019; Lv et al., 2019; Schäfer & Leser, 2020). We follow this approach and include scores returned by classifier c_t in features V_t. We decided to follow a prefix classification approach, which means using separate classifiers c_t for each timestamp t. Following an agreed convention, c.f. Achenchabe, Bondu, Cornuéjols and Dachraoui (2021), Mori, Mendiburu, Keogh, and Lozano (2017), Parrish, Anderson, Gupta, and Hsiao (2013), Schäfer and Leser (2020), Xing et al. (2009, 2011), we perform classification in $T = 20$ timestamps (after 5% of the input sequence, after 10%, 15%, …). As for the classification method, we chose one of the fastest and most accurate methods of Univariate Time Series Classification, MiniROCKET (Dempster et al., 2021).

Transforming classification scores S_t into features for the stopping module V_t can be done in many different ways. We follow the approach presented in TEASER and include all scores as separate elements, followed by the value of the highest score and a difference between the highest and the second-highest score. An exception is made for binary classification, where the score is the only element passed to V_t.

4.3. Learning the classifiers

Each classifier c_t is learned on all prefix subsequences of length t from the training data. Feature extraction using convolutional filters is performed on all samples, and the results are used to fit a Ridge Classifier. In order to choose the best value of the alpha

Algorithm 1 A new solution to Early Time Series Classification

```
Require: T, $X$
  $t \leftarrow 0$                                        ▷ Current timestamp
  while $t < T$ do
      $S_t \leftarrow c_t(X_t)$                           ▷ Scores
      $V_t \leftarrow$ transformToFeatures($S_t$)
      $\hat{C}_\sigma^{(t)} \leftarrow h_t(V_t)$          ▷ Cost difference prediction
      if $\hat{C}_\sigma^{(t)} \geq 0$ then
        stop for those samples
      end if
      $t \leftarrow t + 1$
  end while
  label $\leftarrow$ getLabels($c_t(X_t)$)                ▷ Final classification
  Symbols: ____________________
  $T$ = number of steps, $X_t$ = tested samples at timestamp $t$,
  $c_t$ = classifier dedicated for timestamp $t$,
  $S_t$ = scores calculated at timestamp $t$,
  $V_t$ = features calculated at timestamp $t$,
  $h_t$ = cost difference prediction module dedicated for timestamp $t$,
  $\hat{C}_\sigma^{(t)}$ = cost difference predicted at timestamp $t$
```

parameter (a different parameter than α in the delay cost function), learning the classifier is conducted using a leave-one-out cross-validation. Scores generated during the evaluation of the best alpha parameter are later used to train a calibrator (see Fig. 1). The described learning procedure is also summarized in Algorithm 2.

Algorithm 2 Learning the classifiers

```
Require: T, $X$, $y$
  $t \leftarrow 0$                                        ▷ Current timestamp
  while $t \leq T$ do
      $c_t, S_t \leftarrow$ learn($X_t, y$)               ▷ Learn a classifier, save CV scores
      $V_t \leftarrow$ transformToFeatures($S_t$)
  end while
  return $S, V$
  Symbols: ____________________
  $T$ = number of steps,
  $X_t$ = training samples at timestamp $t$,
  $Y$ = training labels,
  $c_t$ = classifier dedicated for timestamp $t$,
  $S_t$ = scores assigned to samples $X_t$ at timestamp $t$,
  $V_t$ = features assigned to samples $X_t$ at timestamp $t$,
  $V = [V_t]$, $S = [S_t]$
```

4.4. A new approach to stopping

A new method for making the stopping decision performs the future cost prediction, but limits it to only one predicted value. It aims to predict the difference between cost in the case of delaying the prediction for at least one timestamp and cost in the case of halting the process, which is referred to as cost difference.

$$\hat{C}_\sigma^{(t)} = h_t(V_t), \tag{4}$$

where V_t are features computed when time series were available until timestamp t, $\hat{C}_\sigma^{(t)}$ is the expected cost difference, and h_t is a function.

Given this prediction, the stopping criterion can be written as

$$\hat{C}_\sigma^{(t)} \geq 0, \tag{5}$$

which can be interpreted as stopping only if the expected cost if waiting is higher or equal to the expected cost if stopping.

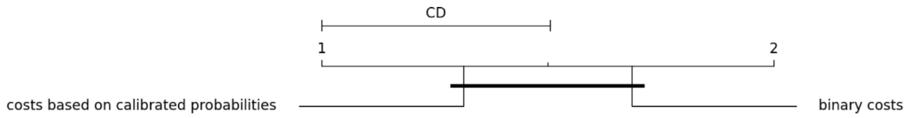

Fig. 3. Critical Difference (CD) diagram for the Nemenyi test showing the results of the statistical comparison of different approaches to cost estimation during training against each other by mean ranks based on the harmonic mean of accuracy and earliness (lower ranks correspond to better results, i.e. higher values of the harmonic mean). Methods that are not connected by a bold line of length equal to CD have significantly different mean ranks (confidence level of 95%). Data consists of training samples from 15 datasets used in Schäfer and Leser (2020). Lower rank is better.

4.5. Generating cost for learning the stopping module

As explained in Section 3.2, the stopping module should learn to predict the moment to stop at the lowest costs A_t based on features V_t. Thus, cost values are needed in the training process. The easiest way to generate them is to set $A_t = C_m(\bar{y}|y) + C_d(t)$, as these would typically be the costs produced by stopping at t. In this paragraph, we show a more complex approach to cost generation that uses classifier calibration in order to apply justified constraints on cost values.

In the beginning, the performance of each classifier c_t is examined by training a calibrator on the scores obtained for cross-validated samples in the previous step. When training is complete, the same scores are passed through the calibrator to calculate probabilities of samples belonging to each class. Please note that this means using the calibrator on the same data it was trained on. As calibration uses a simple logistic regression with a small number of arguments, this technique can create an accurate estimate of the output space. The generated probabilities P are used to estimate the misclassification cost $A_t = P \cdot C_m(\bar{y}|\cdot) + C_d(t)$. For a more formal description of this step, please see Algorithm 3.

Algorithm 3 Generating cost for learning the stopping module

```
Require: T, $S$, $y$
  $t \leftarrow 0$                                        ▷ Current timestamp
  while $t \leq T$ do
      $r_t \leftarrow$ learnCalibrator($S_t$, $y$)
      $P \leftarrow r_t(S_t)$
      $A_t \leftarrow$ probabilitiesToCosts($P$)
      $A_t \leftarrow A_t$+ getDelayCost($t$)
      $t = t + 1$
  end while
  return $A$
  Symbols: ______________________________________________
  $T$ = number of steps,
  $S_t$ = scores assigned to validation samples at timestamp $t$,
  $y$ = validation labels,
  $r_t$ = calibrator dedicated for timestamp $t$,
  $P$ = probabilities of classes estimated for validation samples,
  $A$ = costs estimated for validation samples
```

By using calibration, we ensure that the cost value is modeled as a smooth function of the score. We also introduce well-justified constraints on cost prediction. In order to demonstrate that, let us for a moment consider binary classification. After the estimation of cost values has been carried out using a calibrated classifier, cost as a function of score can have only one local maximum, representing the point where the classifier is the least certain. Thus, scores more distant from this maximum always have lower cost predictions, even if some local fluctuations happen in the data. Thanks to that, the method is less prone to overfitting and should be able to learn better on small quantities of data. We argue that this allows to learn the stopping module better. This assumption was tested on the training data from 15 datasets out of 44 used in Schäfer and Leser (2020). Ten values of alpha were taken from a logarithmic distribution from 0.1 to 10.0, creating ten separate problems for each dataset. The usage of calibrators allowed to achieve better results (see Fig. 3).

4.6. Learning the stopping module

The learning procedure uses calculated misclassification costs a_t to learn a regression method h_t. It starts from $T - 1$, which is the last timestamp where a decision can be made. The cost difference prediction method h_{T-1} learns to predict the cost differences $A_T - A_{T-1}$, where V_{T-1} is an argument. Then, the same method can be used on the training data V_{T-1} to check which samples would trigger a waiting decision. For those that would, the estimated cost values A_{T-1} are swapped with A_T, which the method predicted to be lower. Then, the learning proceeds to the next timestamp and the described steps are repeated. That way, future cost is propagated down the time axis if and only if the sample triggers a waiting decision.

Algorithm 4 Learning the stopping module

```
Require: T, $V$, $A$, $y$
  $t \leftarrow T - 1$
  while $t \geq 0$ do
      $D_t \leftarrow A_{t+1} - A_t$                                          ▷ Cost difference
      $h_t \leftarrow$ learnRegressor$(V_t, D_t)$
      $\hat{C}_{\sigma}^{(t)} \leftarrow h_t(V_t)$                             ▷ Predict response for the training data
      mask $\leftarrow (\hat{C}_{\sigma}^{(t)} < 0)$         ▷ For each element of $V_t$ that would induce waiting
      $A_t \leftarrow$ mask $\cdot A_{t+1} + (1-$mask$) \cdot A_t$              ▷ Simulate waiting
      $t = t - 1$
  end while
  Symbols:
  $T$ = number of steps,
  $V_t$ = features assigned to validation samples at timestamp $t$,
  $A$ = costs estimated for validation samples,
  $h_t$ = cost difference prediction module dedicated for timestamp $t$
```

5. Results

In this section, we present the results of experiments with CALIMERA and comparisons with state-of-the-art methods. Section 5.1 presents the experimental setup and datasets. Section 5.2 compares CALIMERA with ECONOMY. It is contained in a separate subsection, since ECONOMY is applicable only to binary classification problems. Section 5.3 compares CALIMERA with algorithms in multiclass classification problems. The key competitor considered in this section is TEASER.

5.1. Experimental set-up and used datasets

Even though we argue that General ETSC Cost Function is a better-fitted optimization goal for ETSC problems, it is only used when comparing to ECONOMY, which operates on this metric by default. To maintain coherence with authors of the multiclass methods, we use the harmonic mean of accuracy and earliness, the most popular metric able to unify accuracy and earliness. For experiments performed according to the experimental benchmark by Kladis et al. harmonic mean is supplemented by three other popular metrics for quality evaluation.

The experiments concerned publicly available datasets whose properties are given in appendixes in Tables A.3 and B.4. Table A.3 is concerned with datasets used for comparison with ECONOMY, which is limited to binary classification. Table B.4 describes those used for the comparison with multiclass classifiers. For datasets used in the experimental framework by Kladis et al. please refer to Kladis et al. (2021). Please note that these three groups share some datasets.[2]

5.2. Comparison with ECONOMY

This section contains a comparison with the best of the four variants of ECONOMY introduced in Achenchabe, Bondu, Cornuéjols and Dachraoui (2021), ECONOMY-γ, in this paper referred to as ECONOMY. Because ECONOMY is limited to binary classification problems, some multiclass datasets were transformed into binary problems by taking the most numerous class against all others. We worked with 34 datasets in this part of our study. In this comparison, we minimize the General ETSC Cost Function. Both ECONOMY and CALIMERA use it by default. The misclassification cost function C_m is as in Section 3.1. The delay cost function is set to $C_d(t) = \alpha \cdot t$. Experiments were performed on the 27 values of α used in Achenchabe, Bondu, Cornuéjols and Dachraoui (2021). Those are: 0.0001, 0.0002, 0.0004, 0.0008, 0.001, 0.003, 0.005, 0.008, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.

In the original article, the authors decided to list the results of the compared methods for each dataset but for only one alpha value. That was supposed to be the value that "reveals the greatest difference in performance", for which no clarification is given in the paper. We decided to compare ECONOMY and the new algorithm in two separate settings. In the first one, we use the same chosen alpha value for each dataset as in the original article. In the second one, we treat each of the twenty-five α values as separate problems, resulting in mean ranks. This procedure is repeated for each of the 34 datasets.

Achieved cost values are presented in Table 1. Please note that when one method was performing better on a given dataset, it usually kept the advantage for most α values. In only three datasets, the difference between the number of wins was lower than seven, while in 15 datasets, one of the solutions scored more than 21 wins. CALIMERA's advantage was better pronounced for low α; it averaged 28/34 wins for the lowest nine α values, 27.1/34 for the medium values, and only 18.7/34 when the highest nine

[2] In order to download these datasets, one shall visit https://www.timeseriesclassification.com/dataset.php and https://www.dropboxcom/s/4j2em4iwtlhd2sh/deliverable-data.zip?dl=0. The former link was provided by the authors of ECONOMY (Achenchabe, Bondu, Cornuéjols & Lemaire, 2021).

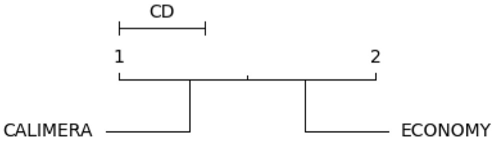

Fig. 4. Mean ranks achieved by ECONOMY and the presented method on 34 datasets and twenty-five α values used in the original evaluation of ECONOMY. Mean ranks are 1.27 and 1.73. Critical difference equals 0.34. Lower rank is better.

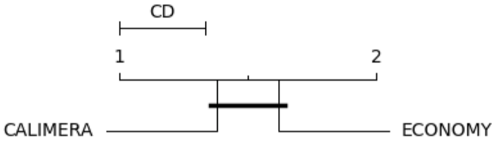

Fig. 5. Mean ranks achieved by ECONOMY and the presented method on 34 datasets and a single α value for each dataset. These values were the same as those that allowed to best highlight differences in performance in the original evaluation of ECONOMY. Mean ranks are 1.38 and 1.62. Critical difference equals 0.34.

Table 1
Comparison of ECONOMY and CALIMERA in two experimental settings described in Section 5.2.

Experimental setting	27 α values		α chosen by authors of ECONOMY		
Metric	Number of wins			Cost	
Dataset	ECONOMY	CALIMERA	α	ECONOMY	CALIMERA
CBF	5	**19**	0.8	**0.186**	0.200
ChlorineConcentration	0	**25**	0.4	0.268	**0.126**
CinCECGTorso	0	**25**	0.1	0.015	**0.007**
Crop	3	**22**	0.06	0.030	**0.023**
ECG5000	7	**18**	0.5	**0.050**	**0.050**
ECGFiveDays	**16**	9	0.3	0.142	**0.098**
ElectricDevices	1	**24**	0.1	0.120	**0.083**
FaceAll	3	**22**	0.01	0.002	**0.001**
FacesUCR	6	**19**	0.5	**0.099**	0.102
FiftyWords	7	**18**	0.5	**0.112**	0.134
FordA	**25**	0	0.3	**0.105**	0.203
FreezerRegularTrain	**25**	0	0.2	**0.016**	0.032
HandOutlines	0	**25**	0.01	0.117	**0.069**
InsectWingbeatSound	1	**24**	1	0.144	**0.135**
ItalyPowerDemand	**16**	9	0.5	0.283	**0.282**
Mallat	12	**13**	0.08	0.017	**0.015**
MedicalImages	1	**24**	0.07	0.232	**0.188**
MelbournePedestrian	**20**	5	0.8	**0.119**	0.134
MixedShapesRegularTrain	0	**25**	0.1	0.025	**0.014**
MoteStrain	9	**16**	0.4	**0.131**	0.139
NonInvasive...Thorax2	0	**25**	0.04	0.011	**0.003**
PhalangesOutlinesCorrect	9	**16**	0.2	**0.288**	0.312
ProximalPhalanx...Correct	0	**25**	0.5	0.314	**0.260**
SemgHandGenderCh2	5	**20**	0.3	0.156	**0.135**
SonyAIBORobotSurface2	2	**23**	0.8	0.211	**0.193**
StarLightCurves	5	**19**	0.3	**0.065**	**0.065**
Strawberry	0	**25**	0.6	0.228	**0.128**
Symbols	**13**	12	0.2	**0.056**	0.069
TwoLeadECG	0	**25**	0.9	0.190	**0.179**
TwoPatterns	7	**18**	0.08	0.082	**0.069**
UWaveGestureLibraryX	11	**14**	0.5	**0.120**	0.132
Wafer	**25**	0	0.2	**0.013**	0.057
WordSynonyms	8	**17**	0.6	**0.174**	0.187
Yoga	0	**25**	0.03	0.131	**0.034**

α values were considered. Thus, CALIMERA's biggest improvement over ECONOMY concerned situations when stopping too early had to be avoided. Results of the Nemenyi test are presented in Fig. 4. Please note that despite the fact that ranks are averaged over $34 \cdot 27 = 918$ tests, the number of problems in the calculation of critical difference was set to 34. The new method performed significantly better than ECONOMY. Results of the Nemenyi test corresponding only to the α parameters chosen in Achenchabe, Bondu, Cornuéjols and Dachraoui (2021) are presented in Fig. 5. The new method performed better. For 20 out of 34 datasets, CALIMERA outperformed ECONOMY. However, in this setting, the difference was not statistically significant.

Table 2
Comparison of the harmonic mean of accuracy and earliness achieved on 45 datasets. The α parameter of CALIMERA was set to 1.0 to roughly mimic maximization of this metric.

Dataset	EARLIEST	EDCS	ECTS	ECDIRE	ECEC	TEASER	CALIMERA
FiftyWords	0.366	0.442	0.366	0.563	0.586	**0.674**	**0.674**
Adiac	0.138	0.155	0.405	0.578	0.715	0.745	**0.770**
Beef	0.231	0.095	0.315	0.390	0.457	0.724	**0.756**
CBF	0.115	0.752	0.421	0.790	0.728	0.620	**0.826**
Chlori...tion	0.148	0.586	0.439	0.678	0.694	**0.696**	0.692
CinCECGtorso	0.048	0.555	**0.926**	0.618	0.831	0.854	0.807
Coffee	0.025	0.570	0.264	0.303	0.860	0.750	**0.905**
CricketX	0.164	0.494	0.373	0.544	0.649	0.690	**0.697**
CricketY	0.212	0.560	0.442	0.635	0.659	**0.720**	0.708
CricketZ	0.220	0.000	0.415	0.568	0.694	**0.721**	0.697
Diatom...tion	0.057	0.785	0.824	0.779	0.860	0.824	**0.889**
ECG200	0.352	0.808	0.552	0.180	**0.880**	0.829	0.876
ECGFiveDays	0.070	0.567	0.456	0.682	0.660	0.724	**0.801**
FaceAll	0.207	0.634	0.489	0.584	0.775	0.836	**0.853**
FaceFour	0.036	0.614	0.417	0.685	0.749	0.713	**0.841**
FacesUCR	0.308	0.545	0.220	0.528	0.717	**0.778**	0.754
Fish	0.106	0.589	0.513	0.579	0.758	0.810	**0.844**
GunPoint	0.065	0.686	0.659	0.763	0.806	0.803	**0.836**
Haptics	0.030	0.489	0.103	0.201	0.506	0.498	**0.595**
InlineSkate	0.021	0.269	0.206	0.373	**0.535**	0.534	0.459
ItalyP...mand	0.158	0.471	0.343	0.454	0.691	**0.721**	0.716
Lightning2	0.039	0.576	0.190	0.678	0.607	**0.744**	0.643
Lightning7	0.164	0.433	0.212	**0.600**	0.555	0.553	0.593
Mallat	0.175	0.595	0.454	0.645	0.640	0.604	**0.675**
MedicalImages	0.501	0.638	0.549	0.764	0.742	0.754	**0.784**
MoteStrain	0.162	0.691	0.339	0.838	0.571	0.837	**0.859**
NonInv...rax1	0.667	0.000	0.346	0.513	0.895	0.900	**0.916**
NonInv...rax2	0.126	0.000	0.365	0.588	0.915	0.920	**0.922**
OliveOil	0.071	0.605	0.227	0.509	0.877	0.890	**0.899**
OSULeaf	0.044	0.505	0.313	0.520	0.711	0.733	**0.756**
SonyAI...ace1	0.149	0.638	0.437	0.521	0.763	0.780	**0.847**
SonyAI...ace2	0.069	0.715	0.588	0.778	0.765	0.785	**0.818**
StarLi...rves	0.026	0.000	0.164	0.629	0.896	0.925	**0.940**
SwedishLeaf	0.273	0.420	0.367	0.666	0.810	0.818	**0.837**
Symbols	0.276	0.448	0.611	0.655	0.751	0.780	**0.819**
Synthe...trol	0.219	0.632	0.211	0.544	0.683	**0.835**	0.827
Trace	0.368	0.692	0.590	0.662	0.746	0.637	**0.797**
TwoPatterns	0.472	0.497	0.226	0.020	0.230	0.556	**0.594**
TwoLeadECG	0.078	0.662	0.482	0.448	0.761	0.780	**0.811**
UWaveG...aryX	0.526	0.432	0.235	0.389	0.540	0.638	**0.706**
UWaveG...aryY	0.521	0.331	0.229	0.058	0.559	0.609	**0.657**
UWaveG...aryZ	0.245	0.446	0.244	0.359	0.572	0.631	**0.683**
Wafer	0.079	0.834	0.715	0.928	**0.970**	0.907	0.942
WordSynonyms	**0.782**	0.395	0.264	0.411	0.518	0.607	0.619
Yoga	0.025	0.656	0.448	0.000	0.759	0.816	**0.858**

5.3. Comparison with multiclass ETSC algorithms

This section compares CALIMERA with other ETSC algorithms capable of solving multiclass classification problems. Among compared methods, we find EDCS, ETSC, ECEC, ECDIRE, EARLIEST and TEASER. In an additional setting, we compare to TSOCF using a subset of 19 datasets chosen by the authors (Zhang & Wan, 2022). It will be shown that the most serious competitor is TEASER, which we address to a greater detail. We use the harmonic mean of accuracy and earliness as the performance metric, because the other algorithms compared in this section favor this optimization criterion. CALIMERA can be roughly adapted to this goal by setting the α parameter to 1.0. When evaluating EARLIEST, 36 combinations of its parameters were tested ($nhid \in \{10, 50\}$, $nlayers \in \{1, 3\}$, $nepochs \in \{20, 100, 500\}$, $lr \in \{0.01, 0.001, 0.0001\}$), and the best result on the test data was chosen.

The results are presented in Table 2. CALIMERA outperforms other methods in most of the considered problems. TEASER, the second best algorithm, was also most similar to CALIMERA, with a Pearson's coefficient 0.87 between the two, followed by ECEC's/CALIMERA 0.82 and TEASER/ECEC 0.79. CALIMERA is able to extrapolate better from a small number of samples than its competitors, as it holds the first place for 13 datasets with the smallest training sets. Please note that, while $\alpha = 1.0$ was used in this experiment, CALIMERA can optimize stopping for different levels of delay penalty. Impact of the α parameter on the accuracy vs earliness trade-off is visualized in Fig. 7.

The CD plot in Fig. 6 summarizes the ranking of algorithms' performance. It was prepared using the Nemenyi test. It shows that CALIMERA outperformed all other multiclass ETSC algorithms. However, the results of the Nemenyi test consider the difference between TEASER and CALIMERA not statistically significant. In the case of all other algorithms, the advantage of CALIMERA was

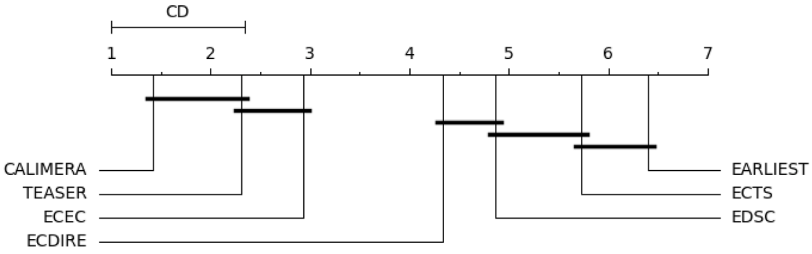

Fig. 6. Mean ranks achieved by methods in the task of maximization of the harmonic mean of accuracy and earliness on 45 datasets. Mean ranks in increasing order: 1.42, 2.31, 2.93, 4.33, 4.87, 5.76, 6.38. Critical difference equals 1.23. Lower rank is better.

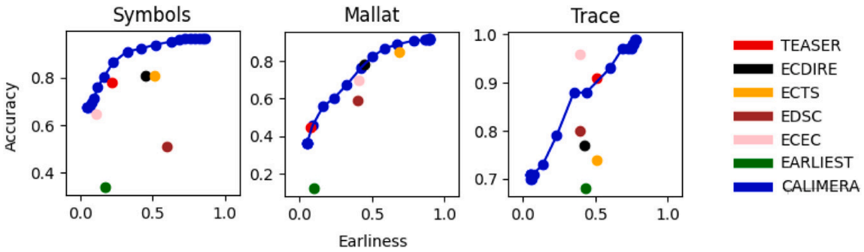

Fig. 7. Comparison of generated accuracy vs earliness trade-offs on three example datasets. Blue points represent results for twenty α values from a log distribution from 0.01 to 10. When α decreased, accuracy and earliness increased. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

clear and statistically significant. To shed more light on the comparison with TEASER, let us mention that for 35 out of 45 datasets CALIMERA achieved better results (c.f. Table 2). The advantage of CALIMERA was not only visible in the number of wins, but also in absolute values of differences of the performance of these two algorithms. Thus, empirical evidence supports our claim that CALIMERA is a well-performing algorithm. There is an additional feature of CALIMERA that is absent in TEASER and it is of a qualitative character. CALIMERA allows the user to specify what is more important: earliness or accuracy.

When tested against TSOCF on a subset of datasets selected by the authors, CALIMERA achieved higher harmonic mean of accuracy and earliness in 16 out of 19 datasets. The remaining three datasets were CBF, Lightning2 and Wafer. Harmonic mean achieved by CALIMERA is presented in Table 2. For exact values of harmonic mean achieved by TSOCF and a list of datasets, please refer to Zhang and Wan (2022). The difference between the two algorithms was statistically significant.

5.4. Comparison using an experimental benchmark

Kladis et al. (2021) created an experimental benchmark to empirically evaluate state-of-the-art ETSC algorithms. They used publicly available data, from which they selected problems well-suited for ETSC, but also introduced two novel datasets. The first dataset (Biological) refers to cancer simulation data, where the population of tumor cells in reaction to administration of specific drugs is simulated in large-scale model experiments. The goal is to classify between effective and ineffective forms of treatment. The second dataset (Maritime) consists of geospatial data of nine vessels, cruising around the port of Brest, France. The goal of classification is to decide whether the examined vessel entered this port.

Kladis et al. compared five solutions considered state-of-the-art. For the purpose of comparison, four metrics were used: accuracy, earliness, harmonic mean of accuracy and earliness and a macro-average of f1-score. For Biological and Maritime, which constituted unbalanced problems (80/20), an additional oversampling step was added to CALIMERA, and α was lowered to 0.3, so instantly selecting the majority class would no longer be the optimal strategy.

The results are presented in Fig. 8. We follow Kladis and al. and present results for the UCR datasets in a single graph. CALIMERA achieved the highest harmonic mean in all three categories. It was also the most early in all of them. Being early meant sacrificing a part of accuracy – in Biological, CALIMERA was the least accurate of all six methods, while in UCR it was the third worst. Performing with much lower earliness and slightly lower accuracy was a profitable strategy – when the only metric that judged the trade-off between accuracy and earliness, harmonic mean, was considered, CALIMERA scored the first place in all three categories. The f1-score achieved in strongly unbalanced problems (Biological and Maritime) confirmed that CALIMERA can be successfully used for unbalanced data.

6. Critical discussion and conclusion

Incorporating calibration into ETSC allows using dedicated methods aimed at estimating probabilities of making a classification error, a task that is always present in some way in ETSC. We strongly believe that future methods of ETSC should use probabilities

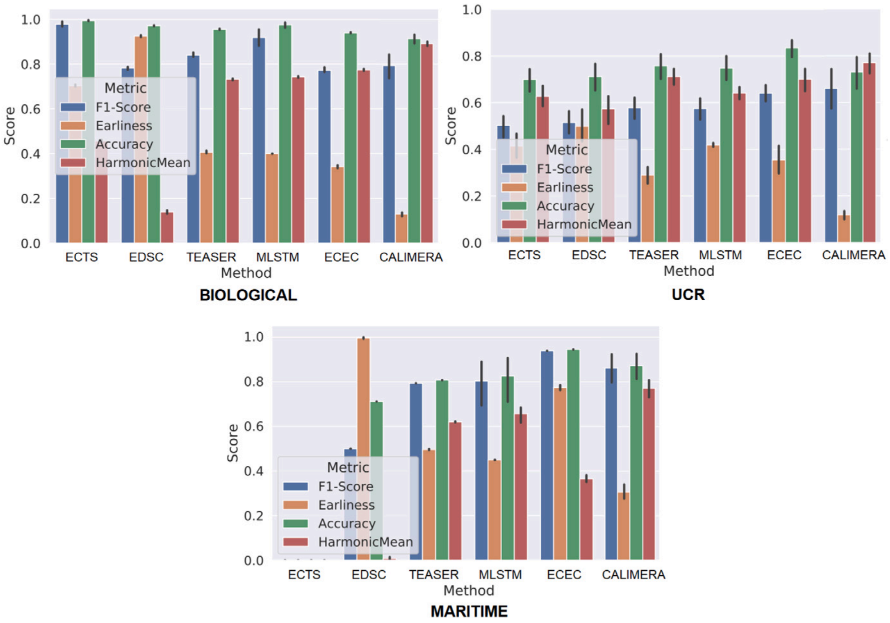

Fig. 8. Results of the comparison using an experimental benchmark.

returned by calibrated classifiers as the default source of information about the misclassification cost. The task of the future methods should not be to estimate the error probabilities (directly or indirectly) but to obtain them from calibrators (treated as black boxes) and use them to predict optimal decisions.

The accuracy of every ETSC method is limited by the classification method it uses. In our opinion, this creates a potential problem for the development of the field. In fact, every time a new best classification method is introduced, one could create a new ETSC method based on it and achieve results better than the current state-of-the-art. In order to overcome this problem, the community could try to evaluate quality of ETSC solutions by separating the performance of the classifier from the performance of the rest of the framework. While it is possible, it would require a very rigorous approach from the researchers – they would have to construct methods so that the classifier could be easily swapped for a different one.

In this paper, we took a step in the direction of the reformulation of the ETSC problem by creating a method that clearly separates the stopping criterion from the rest of the framework. This makes it easy to work on a better stopping method while leaving everything else unchanged. Thanks to the duality of the proposed formulation of the ETSC problem, we see the potential for further development in both emerging areas: decision-making about the early stopping and classifying time series.

Data availability

All used data is publicly available

Acknowledgments

Research was funded by the Warsaw University of Technology within the Excellence Initiative: Research University (IDUB) programme.

Appendix A. Properties of processed datasets for the comparison with ECONOMY

See Table A.3.

Appendix B. Properties of processed datasets for the comparison with multiclass classifiers

See Table B.4.

Table A.3
Elementary information about time series used for comparison with ECONOMY. Data was shared by the authors of ECONOMY, who extracted and modified chosen datasets from http://timeseriesclassification.com. All classification problems are binary.

Dataset name	Length	Train size	Test size	Statistics on train set			Statistics on test set		
				Min	Max	sd	Min	Max	sd
CBF	128	651	279	−3.55	3.77	1.00	−3.04	3.79	1.00
Chlorine...	166	3014	1293	−12.42	12.63	1.00	−9.39	12.59	1.00
CinCECGTorso	1639	994	426	−11.03	11.73	1.00	−11.21	11.67	1.00
Crop	46	16800	7200	0.00	1.05	0.22	0.00	1.01	0.22
ECG5000	140	3500	1500	−7.09	7.40	1.00	−6.28	4.38	1.00
ECGFiveDays	136	618	266	−7.10	6.03	1.00	−7.11	6.02	1.00
ElectricDevices	96	11645	4992	−9.70	9.70	0.99	−9.59	9.70	0.99
FaceAll	131	1575	675	−4.82	9.19	1.00	−4.49	8.88	1.00
FacesUCR	131	1575	675	−4.82	9.19	1.00	−4.58	9.02	1.00
FiftyWords	270	633	272	−2.26	5.28	1.00	−2.52	5.02	1.00
FordA	500	3444	1477	−4.62	5.06	1.00	−4.55	4.27	1.00
FreezerRegularTrain	301	2100	900	−2.23	17.15	1.00	−2.23	8.04	1.00
HandOutlines	2709	959	411	−3.22	2.09	1.00	−2.92	2.08	1.00
Insect...Sound	256	1540	660	−1.30	6.59	1.00	−1.14	6.56	1.00
ItalyPowerDemand	24	767	329	−2.33	3.29	0.98	−2.39	2.80	0.98
Mallat	1024	1680	720	−1.70	2.94	1.00	−1.69	2.90	1.00
MedicalImages	99	798	343	−2.83	8.03	0.99	−2.36	6.93	0.99
MelbournePedestrian	24	2543	1090	−1.00	9682.00	1001.83	−1.00	6661.00	952.44
Mixed...Train	1024	2047	878	−3.28	3.66	1.00	−3.16	3.23	1.00
MoteStrain	84	890	382	−8.64	8.64	0.99	−8.59	8.54	0.99
Non...ECGThorax2	750	2635	1130	−5.42	5.63	1.00	−5.35	4.78	1.00
Phalanges...Correct	80	1860	798	−2.18	2.46	0.99	−2.16	2.26	0.99
Prox...OutlineCorrect	80	623	268	−1.48	1.90	0.99	−1.44	1.85	0.99
Semg...Ch2	1500	466	200	0.01	931.12	20.34	0.02	1933.62	20.64
Sony...Surface2	65	686	294	−4.14	4.50	0.99	−3.92	3.95	0.99
StarLightCurves	1024	6465	2771	−2.62	5.46	1.00	−2.68	5.27	1.00
Strawberry	235	688	295	−2.33	3.72	1.00	−2.10	3.62	1.00
Symbols	398	697	300	−2.59	2.81	1.00	−2.57	2.87	1.00
TwoLeadECG	82	813	349	−3.28	1.93	0.99	−3.80	1.92	0.99
TwoPatterns	128	3500	1500	−1.93	1.94	1.00	−1.94	1.92	1.00
UWave...LibraryX	315	3134	1344	−5.71	4.57	1.00	−5.49	6.51	1.00
Wafer	152	5014	2150	−2.98	12.13	1.00	−3.05	11.95	1.00
WordSynonyms	270	633	272	−2.52	5.28	1.00	−2.26	4.98	1.00
Yoga	426	2310	990	−2.58	2.44	1.00	−2.85	2.41	1.00

Table B.4
Elementary information about time series used for comparison with multiclass classifiers, including TEASER. Data comes from http://timeseriesclassification.com web site.

Dataset name	Classes	Length	Train size	Test size	Statistics on train set			Statistics on test set		
					Min	Max	sd	Min	Max	sd
Adiac	37	176	390	391	−1.99	2.63	1.00	−2.05	2.46	1.00
Beef	5	470	30	30	−3.29	3.72	1.00	−3.39	3.15	1.00
CBF	3	128	30	900	−2.32	3.24	1.00	−3.55	3.79	1.00
ChlorineConcentration	3	166	467	3840	−11.84	7.44	1.00	−12.42	12.63	1.00
CinCECGtorso	4	1639	40	1380	−8.59	10.54	1.00	−11.21	11.73	1.00
Coffee	2	286	28	28	−2.06	2.18	1.00	−2.12	2.10	1.00
CricketX	12	300	390	390	−4.77	11.49	1.00	−5.37	12.65	1.00
CricketY	12	300	390	390	−9.77	6.84	1.00	−10.20	7.41	1.00
CricketZ	12	300	390	390	−4.76	11.92	1.00	−5.13	12.71	1.00
DiatomSizeReduction	4	345	16	306	−1.77	1.98	1.00	−1.98	2.45	1.00
ECG200	2	96	100	100	−2.62	4.20	0.99	−3.01	4.15	0.99
ECGFiveDays	2	136	23	861	−6.51	5.42	1.00	−7.11	6.03	1.00
FaceAll	14	131	560	1690	−4.48	4.88	1.00	−4.82	9.19	1.00
FaceFour	4	350	24	88	−4.69	5.91	1.00	−4.25	5.34	1.00
FacesUCR	14	131	200	2050	−3.96	8.74	1.00	−4.82	9.19	1.00
FiftyWords	50	270	450	455	−2.35	5.02	1.00	−2.52	5.28	1.00
Fish	7	463	175	175	−1.95	2.13	1.00	−1.79	15.05	1.00
GunPoint	2	150	50	150	−2.37	2.05	1.00	−2.50	2.32	1.00

(continued on next page)

Table B.4 (*continued*).

Dataset name	Classes	Length	Train size	Test size	Statistics on train set			Statistics on test set		
					Min	Max	sd	Min	Max	sd
Haptics	5	1092	155	308	−11.15	3.12	1.00	−14.86	3.90	1.00
InlineSkate	7	1882	100	550	−2.26	4.34	1.00	−2.52	3.83	1.00
ItalyPowerDemand	2	24	67	1029	−1.99	2.42	0.98	−2.39	3.29	0.98
Lightning2	2	637	60	61	−1.40	23.13	1.00	−1.45	22.68	1.00
Lightning7	7	319	70	73	−1.78	17.41	1.00	−1.73	16.64	1.00
Mallat	8	1024	55	2345	−1.61	2.76	1.00	−1.70	2.94	1.00
MedicalImages	10	99	381	760	−2.39	7.22	0.99	−2.83	8.03	0.99
MoteStrain	2	84	20	1252	−8.41	2.47	0.99	−8.64	8.54	0.99
NonIn...ECGThorax1	42	750	1800	1965	−5.73	4.79	1.00	−5.75	5.20	1.00
NonIn...ECGThorax2	42	750	1800	1965	−5.42	4.68	1.00	−5.36	5.63	1.00
OliveOil	4	570	30	30	−1.00	3.72	1.00	−1.00	3.73	1.00
OSULeaf	6	427	200	242	−3.16	3.67	1.00	−3.43	3.40	1.00
Sony...Surface1	2	70	20	601	−2.73	3.63	0.99	−3.63	4.00	0.99
Sony...Surface2	2	65	27	953	−4.14	4.23	0.99	−4.02	4.50	0.99
StarLightCurves	3	1024	1000	8236	−2.68	5.29	1.00	−2.62	5.46	1.00
SwedishLeaf	15	128	500	625	−3.41	3.22	1.00	−2.94	3.29	1.00
Symbols	6	398	25	995	−2.31	2.20	1.00	−2.59	2.87	1.00
SyntheticControl	6	60	300	300	−2.45	2.41	0.99	−2.62	2.61	0.99
Trace	4	275	100	100	−2.22	3.97	1.00	−2.39	3.94	1.00
TwoPatterns	4	128	1000	4000	−1.94	1.94	1.00	−1.93	1.92	1.00
TwoLeadECG	2	82	23	1139	−3.15	1.87	0.99	−3.80	1.93	0.99
UWave...LibraryX	8	315	896	3582	−4.44	4.43	1.00	−5.71	6.51	1.00
UWave...LibraryY	8	315	896	3582	−4.10	7.65	1.00	−3.87	5.23	1.00
UWave...LibraryZ	8	315	896	3582	−3.55	4.78	1.00	−4.30	4.86	1.00
Wafer	2	152	1000	6164	−3.05	11.79	1.00	−2.98	12.13	1.00
WordSynonyms	25	270	267	638	−2.26	5.00	1.00	−2.52	5.28	1.00
Yoga	2	426	300	3000	−2.42	2.41	1.00	−2.85	2.44	1.00

References

Achenchabe, Y., Bondu, A., Cornuéjols, A., & Dachraoui, A. (2021). Early classification of time series. *Machine Learning*, *110*(6), 1481–1504. http://dx.doi.org/10.1007/s10994-021-05974-z.

Achenchabe, Y., Bondu, A., Cornuéjols, A., & Lemaire, V. (2021). Early classification of time series is meaningful. arXiv:2104.13257.

Achenchabe, Y., Bondu, A., Cornuejols, A., & Lemaire, V. (2022). Early and revocable time series classification. In *2022 International joint conference on neural networks* (pp. 1–8). http://dx.doi.org/10.1109/IJCNN55064.2022.9892391.

Barandas, M., Folgado, D., Fernandes, L., Santos, S., Abreu, M., Bota, P., et al. (2020). TSFEL: Time series feature extraction library. *SoftwareX*, *11*, Article 100456. http://dx.doi.org/10.1016/j.softx.2020.100456.

Böken, B. (2021). On the appropriateness of Platt scaling in classifier calibration. *Information Systems*, *95*, Article 101641. http://dx.doi.org/10.1016/j.is.2020.101641.

Dachraoui, A., Bondu, A., & Cornuéjols, A. (2015). Early classification of time series as a non myopic sequential decision making problem. In *2015 Joint European conference on machine learning and knowledge discovery in databases* (pp. 433–447). http://dx.doi.org/10.1007/978-3-319-23528-8_27.

Dempster, A., Schmidt, D. F., & Webb, G. I. (2021). MiniRocket: a very fast (Almost) deterministic transform for time series classification. In *27th ACM SIGKDD Conference on knowledge discovery & data mining* (pp. 248–257). http://dx.doi.org/10.1145/3447548.3467231.

Fauvel, K., Balouek-Thomert, D., Melgar, D., Silva, P., Simonet, A., Antoniu, G., et al. (2020). A distributed multi-sensor machine learning approach to earthquake early warning. In *AAAI Conference on artificial intelligence, vol. 34* (pp. 403–411). http://dx.doi.org/10.1609/aaai.v34i01.5376.

Gebel, M. (2009). *Multivariate calibration of classifier scores into the probability space* (Ph.D. thesis), Technische Universität Dortmund, http://dx.doi.org/10.17877/DE290R-863.

Gu, Z.-W., Li, P., Lang, X., Shen, X., Cao, M., & Yang, X.-H. (2021). Hierarchical classification method of electricity consumption industries through TNPE and Bayes. *Measurement and Control*, *54*(3–4), 346–359. http://dx.doi.org/10.1177/0020294021997494.

Gupta, A., Gupta, H. P., Biswas, B., & Dutta, T. (2020). Approaches and applications of early classification of time series: A review. *IEEE Transactions on Artificial Intelligence*, *1*(1), 47–61. http://dx.doi.org/10.1109/tai.2020.3027279.

Gupta, N., Gupta, S. K., Pathak, R. K., Jain, V., Rashidi, P., & Suri, J. S. (2022). Human activity recognition in artificial intelligence framework: a narrative review. *Artificial Intelligence Review*, *55*(6), 4755–4808. http://dx.doi.org/10.1007/s10462-021-10116-x.

Hartvigsen, T., Gerych, W., Thadajarassiri, J., Kong, X., & Rundensteiner, E. (2022). Stop&Hop: Early classification of irregular time series. In *Proc. of the 31st ACM International conference on information & knowledge management* (pp. 696–705). New York, NY, USA: ACM, http://dx.doi.org/10.1145/3511808.3557460.

Hartvigsen, T., Sen, C., Kong, X., & Rundensteiner, E. (2019). Adaptive-halting policy network for early classification. In *25th ACM SIGKDD International conference on knowledge discovery & data mining* (pp. 101–110). http://dx.doi.org/10.1145/3292500.3330974.

Hatami, N., & Chira, C. (2013). Classifiers with a reject option for early time-series classification. In *2013 IEEE Symposium on computational intelligence and ensemble learning* (pp. 9–16). http://dx.doi.org/10.1109/CIEL.2013.6613134.

Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In *2018 IEEE/CVF Conference on computer vision and pattern recognition* (pp. 7132–7141). http://dx.doi.org/10.1109/CVPR.2018.00745.

Huang, Y., Yen, G. G., & Tseng, V. S. (2022a). A novel constraint-based knee-guided neuroevolutionary algorithm for context-specific ECG early classification. *IEEE Journal of Biomedical and Health Informatics*, *26*(11), 5394–5405. http://dx.doi.org/10.1109/JBHI.2022.3199377.

Huang, Y., Yen, G. G., & Tseng, V. S. (2022b). Snippet policy network for multi-class varied-length ECG early classification. *IEEE Transactions on Knowledge and Data Engineering*, 1. http://dx.doi.org/10.1109/TKDE.2022.3160706.

Karim, F., Majumdar, S., Darabi, H., & Harford, S. (2019). Multivariate LSTM-FCNs for time series classification. *Neural Networks*, *116*, 237–245. http://dx.doi.org/10.1016/j.neunet.2019.04.014.

Kladis, E., Akasiadis, C., Michelioudakis, E., Alevizos, E., & Artikis, A. (2021). An empirical evaluation of early time-series classification algorithms. In *EDBT/ICDT Workshops*. URL https://ceur-ws.org/Vol-2841/SIMPLIFY_6.pdf.

Kull, M., Filho, T. S., & Flach, P. (2017). Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In A. Singh, & J. Zhu (Eds.), *Proc. of Machine Learning Research*: *vol. 54*, *Proc. of the 20th International conference on artificial intelligence and statistics* (pp. 623–631). PMLR, URL https://proceedings.mlr.press/v54/kull17a.html.

Lv, J., Hu, X., Li, L., & Li, P. (2019). An effective confidence-based early classification of time series. *IEEE Access*, *7*, 96113–96124. http://dx.doi.org/10.1109/ACCESS.2019.2929644.

Mienye, I. D., & Sun, Y. (2021). Performance analysis of cost-sensitive learning methods with application to imbalanced medical data. *Informatics in Medicine Unlocked*, *25*, Article 100690. http://dx.doi.org/10.1016/j.imu.2021.100690.

Mori, U., Mendiburu, A., Keogh, E., & Lozano, J. A. (2017). Reliable early classification of time series based on discriminating the classes over time. *Data Mining and Knowledge Discovery*, *31*(1), 233–263. http://dx.doi.org/10.1007/s10618-016-0462-1.

Muñoz, M. A., Yan, T., Leal, M. R., Smith-Miles, K., Lorena, A. C., Pappa, G. L., et al. (2021). An instance space analysis of regression problems. *ACM Transactions on Knowledge Discovery in Data*, *15*(2), http://dx.doi.org/10.1145/3436893.

Ottervanger, G., Baratchi, M., & Hoos, H. H. (2021). MultiETSC: automated machine learning for early time series classification. *Data Mining and Knowledge Discovery*, *35*(6), 2602–2654. http://dx.doi.org/10.1007/s10618-021-00781-5.

Paiva, P. Y. A., Moreno, C. C., Smith-Miles, K., Valeriano, M. G., & Lorena, A. C. (2022). Relating instance hardness to classification performance in a dataset: a visual approach. *Machine Learning*, *111*, 3085–3123. http://dx.doi.org/10.1007/s10994-022-06205-9.

Parrish, N., Anderson, H. S., Gupta, M. R., & Hsiao, D. Y. (2013). Classifying with confidence from incomplete information. *Journal of Machine Learning Research*, *14*(76), 3561–3589, URL http://jmlr.org/papers/v14/parrish13a.html.

Platt, J. C. (2000). Probabilities for SV machines. In *Advances in large-margin classifiers* (pp. 61–73).

Ren, H., Wang, J., & Zhao, W. X. (2022). Rsd: a reinforced Siamese network with domain knowledge for early diagnosis. In *31st ACM International conference on information & knowledge management* (pp. 1675–1684). http://dx.doi.org/10.1145/3511808.3557440.

Rußwurm, M., Courty, N., Emonet, R., Lefèvre, S., Tuia, D., & Tavenard, R. (2023). End-to-end learned early classification of time series for in-season crop type mapping. *ISPRS Journal of Photogrammetry and Remote Sensing*, *196*, 445–456. http://dx.doi.org/10.1016/j.isprsjprs.2022.12.016.

Salman, O., Elhajj, I. H., Chehab, A., & Kayssi, A. (2022). Towards efficient real-time traffic classifier: A confidence measure with ensemble deep learning. *Computer Networks*, *204*, Article 108684. http://dx.doi.org/10.1016/j.comnet.2021.108684.

Schäfer, P., & Leser, U. (2020). TEASER: early and accurate time series classification. *Data Mining and Knowledge Discovery*, *34*(5), 1336–1362. http://dx.doi.org/10.1007/s10618-020-00690-z.

Su, W., Jiang, F., Shi, C., Wu, D., Liu, L., Li, S., et al. (2023). An XGBoost-based knowledge tracing model. *International Journal of Computational Intelligence Systems*, *16*(13), http://dx.doi.org/10.1007/s44196-023-00192-y.

Tong, Y., Liu, J., Yu, L., Zhang, L., Sun, L., Li, W., et al. (2022). Technology investigation on time series classification and prediction. *PeerJ Computer Science*, *8*, Article e982. http://dx.doi.org/10.7717/peerj-cs.982.

Van Houdt, G., Mosquera, C., & Nápoles, G. (2020/12/01). A review on the long short-term memory model. *Artificial Intelligence Review*, *53*(8), 5929–5955. http://dx.doi.org/10.1007/s10462-020-09838-1.

Wang, C., Balazs, J., Szarvas, G., Ernst, P., Poddar, L., & Danchenko, P. (2022). Calibrating imbalanced classifiers with focal loss: an empirical study. In *Proc. Of The 2022 Conference on empirical methods in natural language processing* (pp. 145–153). Association for Computational Linguistics, URL https://aclanthology.org/2022.emnlp-industry.14.

Xing, Z., Pei, J., & Yu, P. S. (2009). Early prediction on time series: A nearest neighbor approach. In *21st International joint conference on artificial intelligence* (pp. 1297–1302). URL https://www.ijcai.org/Proceedings/09/Papers/218.pdf.

Xing, Z., Pei, J., Yu, P. S., & Wang, K. (2011). Extracting interpretable features for early classification on time series. In *2011 SIAM International conference on data mining* (pp. 247–258). http://dx.doi.org/10.1137/1.9781611972818.22.

Yao, Y., Chen, H., & Yao, X. (2021). Discriminative learning in the model space for symbolic sequence classification. *The IEEE Transactions on Emerging Topics in Computational Intelligence*, *5*(2), 154–167. http://dx.doi.org/10.1109/TETCI.2019.2914266.

Zhang, W., & Wan, Y. (2022). Early classification of time series based on trend segmentation and optimization cost function. *Applied Intelligence*, *52*, 6782–6793. http://dx.doi.org/10.1007/s10489-021-02788-3.