Early Classification of Time Series: A survey and benchmark

Anonymous authors
Paper under double-blind review

Abstract

In many situations, the measurements of a studied phenomenon are provided sequentially, and the prediction of its class needs to be made as early as possible so as not to incur too high a time penalty, but not too early and risk paying the cost of misclassification. This problem has been particularly studied in the case of time series, and is known as Early Classification of Time Series (ECTS). Although it has been the subject of a growing body of literature, there is still a lack of a systematic, shared evaluation protocol to compare the relative merits of the various existing methods. This document begins by situating these methods within a principle-based taxonomy. It defines dimensions for organizing their evaluation and then reports the results of a very extensive set of experiments along these dimensions involving nine state-of-the-art ECTS algorithms. In addition, these and other experiments can be carried out using an open-source library in which most of the existing ECTS algorithms have been implemented (github available upon release, see attached zip file).

1 Introduction

In hospital emergency rooms (Mathukia et al., 2015), in the control rooms of national or international power grids (Dachraoui et al., 2015), in government councils assessing critical situations, there is a *time pressure* to make early decisions. On the one hand, the longer a decision is delayed, the lower the risk of making the wrong decision, as knowledge of the problem increases with time. On the other hand, late decisions are generally more costly, if only because early decisions allow one to be better prepared. For example, a cyber-attack that is not detected quickly enough gives hackers time to exploit the security flaw found.

A number of applications involve making decisions that optimize a trade-off between the accuracy of the prediction and its earliness. The problem is that favoring one usually works against the other. Greater accuracy comes at the price of waiting for more data. Such a compromise between the *Earliness* and the *Accuracy* of decisions has been particularly studied in the field of Early Classification of Time Series (ECTS) (Gupta et al., 2020), and introduced by Xing et al. (2008). ECTS consists in finding the *optimal time* to trigger the class prediction of an input time series observed over time. Successive measurements provide more and more information about the incoming time series, and ECTS algorithms aim to optimize online the *trade-off* between two conflicting objectives, namely, the earliness and accuracy of class predictions. More formally, we have the following problem.

Problem statement: When deploying an ECTS model, an input time series of size T is progressively observed over time. At time $t \leq T$, the incomplete time series $\mathbf{x}_t = \langle x_1, \ldots, x_t \rangle$ is available where $x_{i(1 \leq i \leq t)}$ denotes the time-indexed measurements. These measurements can be single or multi-valued. The input time series belongs to an unknown class $y \in \mathcal{Y}$. The task is to make a prediction $\hat{y} \in \mathcal{Y}$ about the class of the incoming time series, at a time $\hat{t} \in [1, T]$ before the deadline T. A misclassification cost is incurred when a prediction is made, denoted by $\mathrm{C}_m(\hat{y}|y)$. Furthermore, there exists a delay cost $\mathrm{C}_d(\hat{t})$ that expresses the time pressure and encourages early decisions (defined in Section 2.2). The choice of the best triggering time must optimize a compromise between the two costs, which are moving in opposite directions. As defined in the literature, the ECTS problem involves a training set composed of M labeled time series, denoted by $(\mathbf{x}^i_T, y^i)_{i \in [0,M]} \in (\mathcal{X} \times \mathcal{Y})$, where each series $\mathbf{x}_T = \langle x_1, \ldots, x_T \rangle$ is *complete*, and where each label $y \in \mathcal{Y}$ is associated with the *complete time series*. Consequently, model training and deployment are of different natures. The training stage is carried out as a supervised *batch process*, with access to the full labeled time series. When it comes to testing, on the other hand, decision-triggering is an *online process* which stops at time $\hat{t}$, and at the latest, when the deadline T is reached.

To the best of our knowledge, Alonso González & Diez (2004) are the earliest explicitly mentioning *"classification when only part of the series are presented to the classifier"*. Since then, several researchers have continued their efforts in this direction and have published a large number of research articles. A recent and extensive review of the ECTS approaches can be found in the paper written by Gupta et al. (2020), including the applications that motivated the researchers to work in this area, and covering about fifty relevant papers selected from the 213 papers found by search engines at the time of this writing.

As pointed out by Bondu et al. (2022), the ECTS problem is a special case of optimal stopping (Shepp, 1969; Ferguson, 1989), where the decision to be made concerns both: (*i*) *when* to stop receiving new measurements in order to (*ii*) *predict the class* of the incoming time series.

In the same paper, the ECTS problem has been extended into a more general one, which consists in optimizing the decision times of Machine Learning (ML) models in a wide range of settings where data is collected over time. The authors proposed a set of open research questions to the community, in order to widen the range of applications that are amenable to the ECTS framework (i.e. dealing with other learning tasks, other types of data, other application contexts, etc.).

However, despite the growing interest in ECTS over the last twenty years, there still remains a need for a shared taxonomy of approaches and an agreed well-grounded evaluation methodology. Here, in particular, we list limits that hamper a fair comparison of ECTS methods and algorithms:

1. *Costs taken into account* for evaluating the performance of the proposed method *are not always clearly stated.* It seems natural to distinguish between the misclassification costs $\mathrm{C}_m(\hat{y}|y)$, and the delay cost $\mathrm{C}_d(t)$, and to add them in order to define the cost of making a decision at time t. More generally, the delay cost may depend on the true class y and the predicted one $\hat{y}$, and a single cost function $\mathrm{C}_m(\hat{y}|y,t)$ integrating misclassification and delay costs should then be used. For the sake of clarity, we keep the simple notation that distinguishes both cost functions in the rest of this paper. But in all cases, it is essential to state the framework used and the associated evaluation metric.

2. The performance of the proposed methods should be evaluated *against a range of possible types of cost functions.* It is usual to evaluate "by default" the methods using a ℓ_{0-1} loss function that penalizes a wrong classification by a unity cost, and to consider a linear delay cost function: $\mathrm{C}_d(t) = \lambda\, t$, for a value $\lambda > 0$. However, lots of applications rather involve unbalanced misclassification costs, and possibly also non-linear delay costs. This is the case, for instance, in maintenance applications where wrongly not recognizing a critical situation is much more costly than wrongly predicting a problem and taking steps to fix it, and where delay cost may rise as an exponential function of time: $\mathrm{C}_d(t) = \lambda\, e^t$. It is therefore quite important to assess the adaptability of the methods to various representative problem settings.

3. *The contributions of the various components of a ECTS algorithm should be clearly delineated.* The predominant approach to ECTS is to have a *decision* component which is in charge of evaluating the best moment to make the prediction about the class of the incoming time series, and a *classifier* one which makes the prediction itself. In order to fairly compare the triggering methods, which are at the heart of ECTS, the classifier used should be the same. We call these methods "separable methods".
   An alternative approach relies on having a system that classifies the incoming time series $\mathbf{x}_t$ at each time with a prediction $\hat{y}$ in the set $\{$'*postpone decision*', $y_1, \ldots, y_N\}$ where N is the number of classes. Therefore, within this approach, a single system decides either to wait at least one more time step, or to predict a class and stops the process. In this case, no distinction can be made between a decision component and a classifier one, and the whole system is evaluated as such. In the spirit of deep neural networks, we call this type of method "end-to-end" to underline the fact that a single learning system is in charge of all operations, here *decision* and *classification.* Of course, this precludes a comparison involving only the choice of the decision component with other methods.

4. As with other supervised learning tasks, *performance should be compared with that of "baseline" algorithms.* In the case of ECTS tasks, two naive baselines are: (1) make a prediction as soon as it is allowed, and (2) make a prediction at T, after the entire time series has been observed. In our experiments reported in Section 4, we have added a third baseline, less simple than the two aforementioned ones, but still too obvious so that, to our knowledge, it has never been published as an original method. This is a confidence-based method where a decision is triggered as soon as the confidence for the likeliest class given $\mathbf{x}_t$ is greater than a threshold. (Formally, let $\hat{y} = \arg\max_{y \in \mathcal{Y}} p(y|\mathbf{x}_t)$, then a prediction is made (i.e. $\hat{y}$) as soon as $p(\hat{y}|\mathbf{x}_t) \geq \varepsilon$, for some threshold $\varepsilon \in [0, 1]$.)

5. *Precautions should be taken when using datasets of training time series* to ensure that no bias enters unwillingly into the training and evaluation process. A case in point, concerns the normalization often used in time series datasets. Dau et al. (2019) have reported that 71% of the reference time series classification datasets used to evaluate ECTS methods are made up of z-normalized time series, i.e. with measurements independently modified on each complete series to obtain a mean of zero and a standard deviation equal to 1. Clearly, *this setting is not applicable in practice*, as z-normalization would require knowledge of the entire incoming time series. In a research context, previous work has used such training sets to test the proposed algorithms. As Wu et al. (2021); Achenchabe et al. (2021b) note, this preprocessing is irreversible and can generate a problem for ECTS by introducing a temporal leakage of information. In order to assess its impact, we report in Section C.5 of Appendix C a comparison of results for z-normalized and non-normalized time series.

Up until now, it has been difficult to conduct fair comparisons between competing methods. Often, published performances are based on choices concerning data sets, the precise performance measure used, hyperparameter values and evaluation protocols (e.g., the split between training and test sets) that are not entirely explicit or, in any case, are difficult to reproduce. This is why *we have recoded all the methods*, specified a *shared evaluation protocol* with variants that can be employed by everyone, and searched for a collection of *data sets* that can be widely used to test and compare new as well as existing methods. We hope this will be a useful resource for the scientific community working in this field.

As this community shows a growing interest in the ECTS problem, granted by the increasing number of applications that fall in its range, it is timely (1) to propose a framework into which to cast the various approaches and thus indicate avenues for future research, and (2) a well-grounded evaluation methodology. Specifically, this paper makes the following contributions:

- **A taxonomy** is proposed in Section 2, classifying approaches in the literature according to their design choices.
- **Extensive experiments** have been performed, meeting the above-mentioned shortcomings. **(1)** The experimental protocol in Section 4.1 explicitly defines the costs used during training and evaluation, and varies the balance between misclassification and delay costs by using a large range of cost values. **(2)** Experiments are performed repeatedly for several types of cost function, i.e. balanced or unbalanced misclassification cost, and linear or exponential delay cost (see Sections 4.2 and 4.3) and many intermediate results are available in the supplementary materials. **(3)** Ablation and substitution studies are conducted in Section 4.4 with the aim of evaluating the impact of methodological choices, such as the choice of classifier, its calibration, or even z-normalization of training time series. **(4)** The experiments include three baseline approaches, rarely considered in the literature, which often give surprisingly good results. **(5)** In addition to the reference data used in the ECTS field, a collection of some thirty non-z-normalized datasets is proposed and provided to the community.
- **An open source library** is being made available[1] in order to enable reproducible experiments, as well as facilitate the scientific community's development of future approaches. Particular care has been taken to ensure the quality of the code, so that this library may be used to develop real-life applications.

[1] (see attached zip file)

The rest of this paper is organized as follows : The section 2 proposes and describes a new ECTS taxonomy, the different choices to be made in a well-founded way when designing an ECTS method, and a set of four questions that need to be answered to make these choices.

Section 3 presents a comprehensive view of the ECTS field, along with the suggested taxonomy and the four questions raised in Section 2. In Section 4 we present the pipeline developed in order to realize extensive experimentation and we report the main results obtained for different cost settings. This benchmark is supported by a library released as Open Source for dissemination and used in the ECTS research community. Finally, Section 5 concludes this paper. Appendix B lists the datasets used for the experiments, and complementary results are provided in Appendix C.

2 Organizing the ECTS approaches: a taxonomy

The aim of this section is to outline in a principled way the various choices that need to be made when designing one ECTS method.

General form of an ECTS model

An ECTS approach aims at optimizing a trade-off between accuracy and earliness of the prediction, and thus must be evaluated on this ground. The correctness of the prediction is measured by the misclassification cost $\mathrm{C}_m(\hat{y}|y)$ where $\hat{y}$ is the prediction and y is the true class. The time pressure is sanctioned by a delay cost $\mathrm{C}_d(t)$ that is assumed to be positive and, in most applications, an increasing function of time. We thus consider:

- $\mathrm{C}_m(\hat{y}|y) : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$, that corresponds to the misclassification cost of predicting $\hat{y}$ when the true class is y.
- $\mathrm{C}_d(t) : \mathbb{R}^+ \rightarrow \mathbb{R}$, the delay cost that, usually, is a non-decreasing function over time.

An ECTS function involves a predictor $\hat{y}(\mathbf{x}_t)$, which predicts the class of an input time series $\mathbf{x}_t$ for any $t \in [1, T]$. The cost incurred when a prediction has been triggered at time t is given by a loss function $\mathcal{L}(\hat{y}(\mathbf{x}_t), y, t) = \mathrm{C}_m(\hat{y}(\mathbf{x}_t)|y) + \mathrm{C}_d(t)$. The best decision time t^* is given by:

$$t^\star = \underset{t \in [1,T]}{\arg\min}\, \mathcal{L}(\hat{y}(\mathbf{x}_t), y, t). \tag{1}$$

Let $s^\star \in \mathcal{S}$ an optimal ECTS function belonging to a class of functions $\mathcal{S}$, whose output at time t when receiving $\mathbf{x}_t$ is:

$$s^\star(\mathbf{x}_t) = \begin{cases} \emptyset & \text{if extra measures are queried;} \\ y^\star = \hat{y}(x_{t^\star}) & \text{when prediction is triggered at } t = t^\star; \end{cases} \tag{2}$$

ECTS is however an *online* optimization problem, where at each time step t a function $s(\mathbf{x}_t)$ must decide whether to make a prediction or not. Equation 1 is thus no longer operational since it requires complete knowledge of the time series. In practice, the function $s(\mathbf{x}_t)$ triggers a decision at $\hat{t}$, based on a partial description $\mathbf{x}_{\hat{t}}$ of the incoming time series $\mathbf{x}_T$ (with $t \leq T$). The goal of an ECTS system is to choose a triggering time $\hat{t}$ as close as possible to the optimal one t^*, at least in terms of cost, minimizing $\mathcal{L}(\hat{y}(\mathbf{x}_{\hat{t}}), y, \hat{t}) - \mathcal{L}(\hat{y}(\mathbf{x}_{t^\star}), y, t^\star)$ as much as possible.

From a machine learning point of view, the goal is to find a function $s \in \mathcal{S}$ that best optimizes the loss function $\mathcal{L}$, minimizing the true risk over all time series distributed according to the distribution[2] $\mathbb{P}_{(\mathcal{X} \times \mathcal{Y})}$ that governs the time series in the application:

[2] Notice that the notation $\mathcal{X}$ is an abuse that we use use to simplify our purpose. In all mathematical rigor, the measurements observed successively constitute a family of time-indexed random variables $\mathbf{x} = (\mathbf{x}_t)_{t \in [1,T]}$. This stochastic process $\mathbf{x}$ is not generated as commonly by a distribution, but by a filtration $\mathbb{F} = (\mathcal{F}_t)_{t \in [1,T]}$ which is defined as a collection of nested σ-algebras (Klenke, 2013) allowing to consider time dependencies. Therefore, the distribution $\mathcal{X}$ should also be re-written as a filtration.

$$\arg\min_{s \in \mathcal{S}} \mathbb{E}_{(\mathbf{x},y)\sim\mathbb{P}_{(\mathcal{X}\times\mathcal{Y})}} \left[\mathcal{L}(\hat{y}(\mathbf{x}_{\hat{t}}), y, \hat{t})\right] \tag{3}$$

The questions then are:

1. Which form can take the function $s(\cdot)$? We will distinguish *end-to-end* architecture from *separable* ones.
2. How the *criterion* to be optimized accounts for the trade-off between accuracy and earliness? We will see that the costs implied, about misclassification and delay, are variously explicit in the existing methods.
3. How the *when question* of the stopping problem can be approached? This will lead us to distinguish between *cost-informed* and *cost-uninformed* ones on one hand, and between *anticipation-based* methods versus *myopic* ones, on the other hand.
4. How the prediction problem itself can be solved given that $\mathbf{x}_t$ belongs to a *different input spaces* at each time step t?

This set of questions and possible choices for their solution are illustrated in Figure 1. We turn successively to each one in the following.

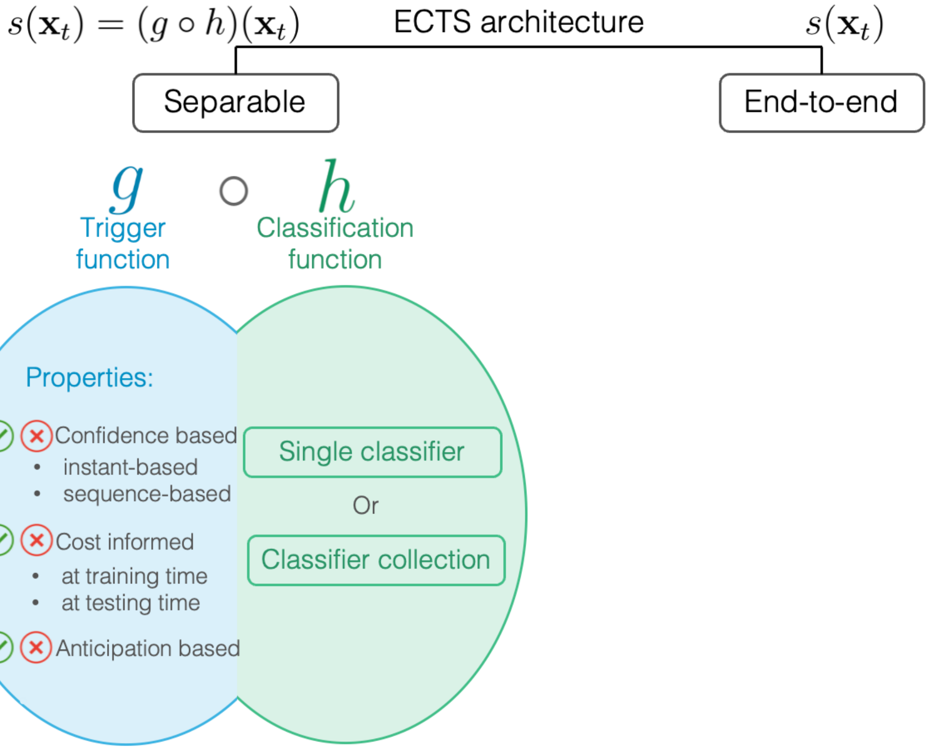


Figure 1: Proposed ECTS taxonomy

2.1 The different forms of the function $s(\cdot)$

An ECTS function must solve both the question of (i) *when to stop* receiving new measurements and decide to make a prediction and (ii) *how to make the prediction* about the class of the incoming time series $\mathbf{x}_t$.

In the *separable approach*, these questions are solved using two separate components. The *classification* one deals with making a prediction: $\mathbf{x}_t \mapsto \hat{y}$, while the *trigger* function decides when to predict. Within this perspective, the classification component is learned independently of the trigger one, while the latter uses the results of the classification component in order to trigger a decision. A simple triggering strategy is to decide it is time to make a prediction as soon as the classification component is sufficiently sure of its prediction. We formalize separable approaches by: $s(\mathbf{x}_t) = (g \circ h)(\mathbf{x}_t)$ where g is the decision or trigger function, and h is the prediction function.

In the *end-to-end* approaches, a single component decides when to make a prediction and what that prediction is. Thus, the function s, defined in Equation 2, is responsible both for choosing the time $\hat{t}$ for making the predictions, and for the prediction itself $\hat{y}$.

The question that naturally arises is which type of architecture (i.e. end-to-end or separable) performs best. On the one hand, in separable approaches, the classification component is trained independently of the triggering one, which can be detrimental, for example by propagating errors from one module to another. On the other hand, separating the ECTS problem into two inherently simpler sub-problems could be an advantage. In this paper, we do not delve any further into this question, which we leave for future work.

2.2 Choice of the optimizing criterion *during training*

This section covers optimization criteria existing in the literature and used to train ECTS functions. In the following, these criteria are listed by level of cost awareness, and we differentiate between the following situations:

1. The first one, we call *cost-informed at training time.*

   $\mathbb{P}_{(\mathcal{X}\times\mathcal{Y})}$ being unknown, instead of using Equation 3 describing the true risk, one tries to minimize the *empirical risk*, also called *average cost* in the ECTS literature, for a training set of M time series:

   $$AvgCost \;=\frac{1}{M}\sum_{i=1}^{M}\mathcal{L}(\hat{y}_i, y_i, \hat{t}) \;=\; \frac{1}{M}\sum_{i=1}^{M}\mathrm{C}_m(\hat{y}_i|y_i) + \mathrm{C}_d(\hat{t}_i) \tag{4}$$

   AvgCost is the most appropriate criterion to both train and evaluate ECTS approaches, as this is what will eventually be paid by a practitioner. The following presents proxy measures of the *empirical risk.*

2. The second situation is a sub-case of cost-informed, and can be qualified as *cost-proxy at training time.* There, while the accuracy and earliness of prediction are taken into account, the optimization criterion combines them in a proxy which is not the *AvgCost.* Classical proxies include:

   - The *Harmonic Mean* (see Schäfer & Leser (2020)):

     $$HM = \frac{2 \times \mathit{Accuracy} \times (1 - \mathit{Earliness})}{\mathit{Accuracy} + (1 - \mathit{Earliness})} \tag{5}$$

     with:

     $$\mathit{Accuracy} = \frac{1}{M}\sum_{i=1}^{M}\mathbb{1}(\hat{y}_i = y_i) \tag{6}$$

     $$\mathit{Earliness} = \frac{1}{M \times T}\sum_{i=1}^{M}\hat{t}_i \tag{7}$$

     where $\hat{t}_i$ is the time that the ECTS function decides to make prediction for the time series $\mathbf{x}^i$.

   - The *CF* (i.e. Cost Function) criterion with $0 \leq \alpha \leq 1$ (see Mori et al. (2017a); Lv et al. (2019; 2023)):

     $$CF = \alpha \times (1 - \mathit{Accuracy}) + (1 - \alpha) \times \mathit{Earliness} \tag{8}$$

     When the cost of misclassification $\mathrm{C}_m(\hat{y}, y) = \mathbb{1}(\hat{y} = y)$, and the delay cost is $\mathrm{C}_d(t) = \hat{t}/T$ and $\alpha = 0.5$, CF becomes a particular case of *AvgCost.*

The choice of using costs or approximating them by a proxy is a technical issue, which may, for example, be relevant to making the loss function differentiable by approximation. For this reason, in the remainder of this paper, *cost-informed/cost-proxies at training time* situations are grouped together under the term *cost-informed at training time*, this difference not being essential.

3. The third situation is qualified as *cost-uninformed-train.* This is the case of methods that only set the threshold on the confidence of the prediction in order to trigger the prediction, regardless of the costs incurred on the training set. Another example is methods that use rigid rules, such as "decide as early as the first measurement is available".

It is a question whether any of these approaches fare better than the others. To measure this, we must use the *Average Cost* (see Equation 4) measured on a test set, which represents the ground truth of the ECTS problem. It should be noted that many of the proposed methods have been evaluated on the basis of other criteria in the literature, with the result of not allowing a rigorous comparison between them. We will come to this problem in Section 4.

The rest of this section is specific to *separable* ECTS approaches, which represent a large part of the literature.

2.3 Information used by the trigger function *during inference*

The trigger function can draw on different types of information. In the simplest case, it can decide irrespective of the incoming time series $\mathbf{x}_t$. This is the case, for example, of the rule that would say: "*wait until half of the measurements are available, then make a prediction*". The corresponding trigger function can be said to be blind.

Apart from this extreme case, it is interesting to distinguish *two dimensions.* First, the trigger function may or may not take the misclassification and delay costs into account. Second, it can also make its decision at each time step solely on the basis of past observations, or it can anticipate possible futures to help its decision.

For instance, *confidence-based* methods (see Section 2.3.1) do not explicitly take costs into account in the triggering decision.

2.3.1 Confidence-based approaches

Confidence-based approaches are widely used in the literature. The simplest trigger model of this kind consists in monitoring a quantity related to the confidence of the prediction over time and triggering class prediction as soon as a threshold value is exceeded. The confidence metric monitored can take different forms. For example, a baseline approach, referred to as *Proba Threshold*[3] in the remainder of this paper, involves monitoring $\max_{y \in \mathcal{Y}} p(y|\mathbf{x}_t)$ the highest conditional probability estimated by the classifier. This baseline example is qualified as *instant-based* method, since it takes as input only the last confidence score available at time t. Another type of approaches, qualified as *sequence-based*, monitors the entire sequence of past confidence scores, and triggering a prediction is made conditionally on a particular property of this sequence. Accordingly, trigger functions can either take as input a scalar value, e.g. $g(\max_{y \in \mathcal{Y}} p(y|\mathbf{x}_t))$, in the case of *instant-based* approaches, or a sequence of scalar values, e.g. $g(\{\max_{y \in \mathcal{Y}} p(y|\mathbf{x}_\tau)\}_{1 \leq \tau \leq t})$, in the case of *sequence-based* approaches (see Section 3.1).

2.3.2 Cost-informed at testing time

Given that an ECTS approach will ultimately be evaluated on the average cost of using it (see Equation 4), it seems natural to exploit the cost values at testing time, in order to trigger predictions at optimal moments. Methods such as Economy (Achenchabe et al., 2021a) and 2step/NoCluster (Tavenard & Malinowski, 2016) do that. They can thus be qualified as "*cost-informed at testing time*".

Other approaches use instead trigger functions that, once learned, do not take into account the cost at testing time, but rely on other measures such as, for instance, the confidence of the prediction. This is the case of

[3] Baseline implemented in the aeon (Middlehurst et al., 2024a) library : `https://urlz.fr/qmWl`

the SR approach (Mori et al., 2017a). Therefore, these approaches can be qualified as "*cost-uninformed at testing time*".

Notice that some approaches are cost-informed during training but not during inference. This is the case with the SR approach, which is *cost-informed at training time* since it uses costs to optimize its parameters, and *cost-uninformed at testing time* since the resulting trigger function does not use cost values during inference. Table 1 shows these two different properties for each approach in the literature.

2.3.3 Anticipation-based decisions

Some separable approaches consider the output of the classifier h at time t to decide whether this is the right time to make a prediction. For instance, stopping the process when the confidence in the classification $h_t(\mathbf{x}_t)$ is above some threshold, or when the difference of confidence between the best two predictions exceeds some value. These methods can be described as *myopic* since they only look at the current time step t, without trying to guess the future.

But there is another possibility. As was first noted by Achenchabe et al. (2021a), the ECTS problem can be cast as a LUPI (Learning Using Privileged Information) problem (Vapnik & Vashist, 2009). In this scenario, the learner can benefit at the training time from privileged information that will not be available at test time. Formally, the training set can be expressed as $\mathcal{T} = \{(\mathbf{x}_i, \mathbf{x}_i^\star, y_i)\}$, where $\mathbf{x}_i$ is what is observable and $\mathbf{x}_i^\star$ is some additional information not available when the prediction must be made. This is exactly what happens in the ECTS problem. Whereas at test time, only $\mathbf{x}_t$ is available, during training the complete time series is known. This brings the possibility to learn what are the likely futures of an incoming time series $\mathbf{x}_t$ provided it comes from the same distribution. Hence, it becomes also possible to guess the cost to be optimized for all future time steps, and therefore to wait until the moment seems the best. This type of approach can be said *anticipation-based* (also called *non-myopic* in the literature). Because more information from the training set is exploited, it can be expected that these methods outperform *myopic* and *blind* ones.

Is this confirmed by experience? Are there situations where the advantage is significant? Our experiments in Section 4 provide answers to these questions.

2.4 Choice of the classification component

One source of difficulty when devising an ECTS method in the separable setting is that, at testing time, inputs differ from one time step to another. When an incoming time series is progressively observed, the number of measurements, and hence the input dimension, varies. Two approaches have been used to deal with the problem.

1. A *set of classifiers* $\{h_t\}_{t \in [1,T]}$ is learned, each dedicated to a given time step t, and thus a given input dimension. In practice, authors often choose a limited subset of timestamps, usually a set of twenty (one measurement every 5% of the length of the time series), to restrict the number of classifiers to learn and therefore the associated computational cost.

2. A *single classifier* h is used for all possible incoming time series $\mathbf{x}_t$. One way of doing this is to "project" an input $\mathbf{x}_t$ of dimension $t \times d$, if d is the dimension of an observation at time t (i.e. multi-valued time series), into a fixed dimensional vector whatever t and d. This may simply be the mean value and standard deviation of the available measurements (multiplied by the dimension d) or the result of a more sophisticated feature engineering as tested by Skakun et al. (2017). Deep learning architectures can also be used to learn an encoding of the time series in an intermediate layer. For instance, Wang et al. (2016); Sawada et al. (2022) use a CNN architecture, and Lv et al. (2023) a FCN one. Wang et al. (2017) show that using deep neural architectures often performs well for time series classification.

Both approaches have their own limitations. On the one hand, using a set of classifiers, each independently dedicated to a time step, does not exploit information sharing. On the other hand, using a single classifier seems to be a more difficult task, as the representation of $\mathbf{x}_t$ can be different at times t and $t + 1$ and all

Table 1: Table of published methods for the ECTS problem with their properties along dimensions underlined in the taxonomy.

References	Classifier(s) (collection ✓)	End2end	Confidence	Anticipation	Cost informed
EDSC (Xing et al., 2011)	Shapelet	✓	instant		
ECTS (Xing et al., 2012)	1NN		instant	✓	
Reject (Hatami & Chira, 2013)	SVM		instant		
RelClass (Parrish et al., 2013)	QDA, Linear SVM		instant	✓	
iHMM (Antonucci et al., 2015)	HMM		instant		
2step/NoCluster (Tavenard & Malinowski, 2016)	Linear SVM (✓)			✓	train & test
ECDIRE (Mori et al., 2017b)	Gaussian Process (✓)		instant		
Stopping Rule (Mori et al., 2017a)	Gaussian Process (✓)		instant		train
EARLIEST (Hartvigsen et al., 2019)	LSTM				train
ECEC (Lv et al., 2019)	WEASEL (✓)		sequence		train
DDQN (Martinez et al., 2020)	MLP	✓		✓	train
TEASER (Schäfer & Leser, 2020)	WEASEL (✓)		sequence		train
ECONOMY-γ-max (Zafar et al., 2021)	XGBoost + tsfel (✓)			✓	train & test
DETSCNet (Chen et al., 2022)	TCN	✓			train
Benefitter (Shekhar et al., 2023)	LSTM	✓		✓	train
CALIMERA (Bilski & Jastrzebska, 2023)	MiniROCKET (✓)			✓	train
ELECTS (Rußwurm et al., 2023)	LSTM	✓			train
SOCN (Lv et al., 2023)	FCN		sequence		train
EarlyStop-RL (Wang et al., 2024)	MLP	✓		✓	train
FIRMBOUND (Ebihara et al., 2025)	MLP		instant	✓	train

further time steps which can lead to additional difficulty for the classifier while moreover requiring a more demanding feature engineering step. Therefore, here also, it is interesting to measure experimentally whether one dominates the other. This will be the subject of future work.

3 State of the art

Literature approaches can be mapped within the proposed taxonomy, as shown in Table 1. In this section, literature approaches are considered focusing on their core ideas and main contributions. This section is organized around two key notions from the previously introduced taxonomy: *confidence-based* and *anticipation-based*. The remainder of this section is organized as follows. Subsection 3.1 presents methods whose decisions are triggered in a myopic way, based on some confidence measure. Subsection 3.2 describes approaches using a non-myopic decision criterion, which attempts to anticipate likely continuations. Subsection 3.3 discusses RL-based approaches, which although anticipation-based, are rather singular in their different formalism. Finally, Subsection 3.4 presents Deep Learning-based approaches, which are not much developed in the literature so far.

3.1 Confidence-based approaches

Most ECTS methods to date are *separable*, *confidence-based*, *cost-informed at training time*, and are not *anticipation-based*. They implement separately the prediction and the triggering components, they learn them using the costs, hence they are *cost-informed at training time*, but they decide to trigger a decision based on the information available at the current time step t without trying to anticipate the likely future, and they base their decision upon the confidence of the predictions made by the classifier.

There exist two families of confidence-based approaches. In the *first* one, only the last time step is considered, a score based on confidence estimations is monitored at each time step and a class prediction is triggered as soon as a threshold on this score is exceeded. By contrast, in the *second*, a sequence of estimated scores is monitored, and the condition to trigger a decision depends upon some property of this sequence.

3.1.1 Instant-based decision criterion

• One basic method is to monitor $\max_{y \in \mathcal{Y}} p(y|\mathbf{x}_t)$, the highest conditional probability estimated by the classifier, which is a simple measure of classifier confidence over time. As soon as it exceeds a value, which is a hyperparameter of the method, a prediction is made. We call this method Proba Threshold and use it as a baseline for comparison later in our experiments.

• The Reject method (Hatami & Chira, 2013) uses ensemble consensus as a confidence measure. For each time step, first (*i*), a pool of classifiers is trained by varying their hyperparameters (i.e. SVMs); then (*ii*), the most accurate of these are selected; and (*iii*) the pair of classifiers minimizing their agreement in predictions is chosen to form the ensemble. Finally, the prediction is triggered as soon as both classifiers in the ensemble predict the same class value. In this case, the monitored confidence measure is binary (agreement or disagreement), there is no trigger threshold and thus this trigger model is free of hyperparameters[4].

• Hidden Markov Models (HMMs) are naturally suited to the classification of online sequences. An HMM is learned for each class, and at each time step t, the class to be preferred is the one with the highest a posteriori probability given $\mathbf{x}_t$. However, the decision to make a prediction now or to postpone it must then involve a threshold so that the prediction is only made if the a posteriori probability of the best HMM is sufficiently high or is greater than that of the second-best. In reaction to this, Antonucci et al. (2015) propose to replace the standard HMM with *imprecise HMMs* based on the concept of credal classification. This eliminates the need to choose a threshold, since a decision is made when one classification “dominates” (according to a criterion based on probability intervals) all the others.

• Rather than considering only the largest value predicted by the classifier, it is appealing to consider also the difference with the second largest value, since a large difference points to the fact that there is no tie between predictions to expect.

This is one dimension used in the Stopping Rule (SR) approach (Mori et al., 2017a). Specifically, the output of the system is defined as:

$$g(h(\mathbf{x}_t)) = \begin{cases} \emptyset & \text{if extra measures are queried;} \\ \hat{y} = \arg\max_{y \in \mathcal{Y}} p(y|\mathbf{x}_t) & \text{when } \gamma_1 \, p_1 + \gamma_2 \, p_2 + \gamma_3 \, \frac{t}{T} > 0 \end{cases} \tag{9}$$

where p_1 is the largest posterior probability $p(y|\mathbf{x}_t)$ estimated by the classifier h, p_2 is the difference between the two largest posterior probabilities, and $\frac{t}{T}$ represents the proportion of the incoming time series at time t. The parameters γ_1, γ_2 and γ_3 are learned from the training set.

• Using the same notations as SR, the Early Classification framework based on class DIscriminativeness and RELiability (Ecdire) (Mori et al., 2017b) finds the earliest timestamp for which a threshold applied on p_1 is reached (defined as in Equation 9). Then, the quantity p_2 is monitored, and a second threshold is applied to trigger the prediction.

• A different class of methods relies on searching telltale representations of subsequences, such that if the incoming time sequence $\mathbf{x}_t$ matches one or more of these representations, then its class can be predicted. Typically, these representations take the form of shapelets that discriminate well one class from the others (Ye & Keogh, 2011). For instance, the Early Distinctive Shapelet Classification (Edsc) method learns a distance threshold for each shapelet, based on the computation of the Euclidean distance between the considered subsequence and all other valid subsequences in the training set (Xing et al., 2011). It selects a subset of them, based on a utility measure that combines precision and recall, weighted by the earliness. A prediction is made as soon as $\mathbf{x}_t$ matches one of these shapelets well enough. Because this family of methods is computationally expensive, extensions have been developed to reduce the computational load (Yan et al.,

[4] The Reject approach involves choosing the number of classifiers trained in step (*i*) and selected in step (*ii*) that could be considered as hyper-parameters of the monitored confidence measure.

2020; Zhang & Wan, 2022). Other extensions aim at improving the reliability of the predictions (Ghalwash et al., 2014; Yao et al., 2019), and tackling multivariate time series (Ghalwash & Obradovic, 2012; He et al., 2013; 2015; Lin et al., 2015).

• Ringel et al. (2024) use the *Learning Then Test* (LTT) (Angelopoulos et al., 2021) calibration framework to address ECTS. In practice, the proposed approach greedily computes thresholds at each time step, in order to monitor some conditional control risk measure, given a pre-defined error rate. This paper investigates text applications.

• Historically, a related scenario predates the ECTS problem but is different. In the *sequential decision making* and *optimal statistical decisions* frameworks (DeGroot, 2005; Berger, 1985), the successive measurements are supposed to be independently and identically distributed (i.i.d.) according to a distribution of unknown "parameter" θ. The problem is to determine as soon as possible whether the measurements have been generated by a distribution of parameter θ_0 (hypothesis H_0) or of parameter θ_1 (hypothesis H_1) with $\theta_0 \neq \theta_1$. In the Wald's Sequential Probability Ratio Test (Wald & Wolfowitz, 1948; Ghosh & Sen, 1991), the log-likelihood ratio $R_t = \log \frac{P(\langle x_1^i,\ldots,x_t^i\rangle \mid y=-1)}{P(\langle x_1^i,\ldots,x_t^i\rangle \mid y=+1)}$ is computed and compared with two thresholds that are set according to the required error of the first kind α (*false positive error*) and error of the second kind β (*false negative error*). This beautiful setting allows one to get optimal decision times at the cost of being able to compute the log-likelihood. However, it differs from the ECTS problem, where successive observations are dependent. The i.i.d. assumption being not valid for the ECTS problem, a generalization to the non-i.i.d. case was proposed by Tartakovsky et al. (2014), providing guarantees for the asymptotic case (with $T \to \infty$). Despite this latter limitation, Ebihara et al. (2025) has recently applied this type of approach to ECTS with finite time horizons. The authors propose practical ways of both estimating R_t (Ebihara et al., 2023) and triggering times by solving a backward induction problem (Tartakovsky et al., 2014).

3.1.2 Sequence-based decision criterion

Other approaches propose *sequence-based* confidence measures specifically designed for the ECTS problem.

• The Effective Confidence-based Early Classification (Ecec) (Lv et al., 2019) proposes a confidence measure based on the sequence of predicted class values, from the first one observed to the current timestamp. At each time step, this approach exploits the *precision* of the classifier to estimate the probability for each possible class value $y \in \mathcal{Y}$ of being correct if predicted. Then, assuming that successive class predictions are independent, the proposed confidence measure represents the probability that the last class prediction is correct given the sequence of predicted class values. The proposed confidence measure is monitored over time, and prediction is triggered if this measure exceeds a certain threshold γ tuned as the single hyperparameter.

• The Teaser (Two-tier Early and Accurate Series classifiER) (Schäfer & Leser, 2020) approach considers the problem of whether or not a prediction should be triggered as a classification task, the aim of which is to discriminate between *correct* and *bad* class predictions. As the authors point out, the balance of this classification task varies according to the time step considered $t \in [1, T]$. Indeed, assuming there is an information gain over time, there are fewer and fewer bad decisions as new measurements are received (or even no bad decisions after a while, i.e. $\forall\ t > t'$ $(0 < t' \leq T)$ for some datasets). To exploit this idea, a collection of one-class SVMs is used, learning hyper-spheres around the correct predictions for each time step. A prediction is triggered when it falls within these hyper-spheres for ν consecutive time steps (ν being a parameter of the method).

• The Second-Order Confidence Network approach (Socn) (Lv et al., 2023) considers, as does Teaser, the same classification task aiming to discriminate between correct and bad predictions. To learn this task, a transformer (Vaswani et al., 2017) is used, taking as input the complete sequence of conditional probabilities estimated by the classifier h, from the first time step, up to the current time step. A confidence threshold ν is learned by minimizing the same cost function as Lv et al. (2019) do, above which the prediction is considered reliable and therefore triggered.

3.2 Anticipation-based methods

• One way of designing approaches that anticipate future measurements is to achieve classification of an incomplete time series while guaranteeing a minimum probability threshold according to which the same decision would be made on the complete series. This is the case of the Reliability Classification (RelClass) approach (Parrish et al., 2013). Assuming that the measurements are i.i.d. and generated by a Gaussian process, this approach estimates $p(\mathbf{x}_T|\mathbf{x}_t)$ the conditional probability of the entire time series $\mathbf{x}_T$ given an incomplete realization $\mathbf{x}_t$ and thus derives guarantees of the form:

$$p\big(h_T(\mathbf{x}_T) = y|\mathbf{x}_t\big) \;=\; \int_{\mathbf{x}_T \text{ s.t. } h_T(\mathbf{x}_T)=y} p(\mathbf{x}_T|\mathbf{x}_t)\, d\mathbf{x}_T \;\geq\; \gamma$$

where $\mathbf{x}_T$ is a random variable associated with the complete time series, γ is a confidence threshold, and h_T is the classifier learned over the complete time series. At each time step t, $p(h_T(\mathbf{x}_T) = y|\mathbf{x}_t)$ is evaluated and a prediction is triggered if this term becomes greater than the threshold γ, which is the only hyper-parameter to be tuned.

• Another way of implementing anticipation-based approaches is to exploit the continuations of training time series, which are full-length. One of the first methods for ECTS has been derived into such an anticipation-based approach. The first, called Early Classification on Time Series (Ects) (Xing et al., 2009), exploits the concept of Minimum Prediction Length (MPL), defined as the earliest time step for which the predicted label should not change for the incoming time series $\mathbf{x}_t$ from t to T. This is estimated by looking for the 1NN of $\mathbf{x}_t$ in the training set, and checks whether from t onward, its predicted label did not change. To be more robust, the MPL is defined based on clusters computed on full-length training time series to estimate the best decision time. The approach has been extended later on to speed up the learning stage (Xing et al., 2012). This method looks in its own way at the likely future of $\mathbf{x}_t$ - i.e. an incomplete time series belongs to a cluster whose continuations are known - and thus can be considered as an anticipation-based method.

• Dachraoui et al. (2015) present a method that claims explicitly to be “non-myopic” in that a decision is taken at time t only insofar as it seems that no better time for prediction is to be expected in the future. In order to do this, the family of Economy methods estimates the future cost expectation based on the incoming time series $\mathbf{x}_t$. This can be done since the training data consists of full-length time series and therefore a Learning Using Privileged Information (LUPI) (Vapnik & Vashist, 2009) is possible.

More formally, the objective is to trigger a decision when $\mathbb{E}_{y,\hat{y}}[\mathcal{L}(\hat{y}, y, t)|\mathbf{x}_t]$ is minimal, with:

$$\mathbb{E}_{y,\hat{y}}[\mathcal{L}(\hat{y}, y, t)|\mathbf{x}_t] = \sum_{y \in \mathcal{Y}} P(y|\mathbf{x}_t) \sum_{\hat{y} \in \mathcal{Y}} P(\hat{y}|y, \mathbf{x}_t)\, C_m(\hat{y}|y) \;+\; C_d(t) \tag{10}$$

A tractable version of Equation 10 has been proposed by introducing an additional random variable which is the membership of $\mathbf{x}_t$ to the groups of a partition $\mathcal{G}$:

$$\mathbb{E}_{y,\hat{y}}[\mathcal{L}(\hat{y}, y, t)|\mathbf{x}_t] = \sum_{g_k \in \mathcal{G}} P(g_k|\mathbf{x}_t) \sum_{y \in \mathcal{Y}} P(y|g_k) \sum_{\hat{y} \in \mathcal{Y}} P(\hat{y}|y, g_k) C_m(\hat{y}|y) + C_d(t) \tag{11}$$

In technical terms, training approaches from the Economy framework involve estimating the three probability terms of Equation 11, for the current time step t, as well as for future time steps $t + \tau \in [t + 1, T]$, with:

- $P(g_k|\mathbf{x}_t)$ the probability of $\mathbf{x}_t$ belonging to the groups $g_k \in \mathcal{G}$,
- $P(y|g_k)$ the prior probability of classes in each group,
- $P(\hat{y}|y, g_k)$ the probability of predicting $\hat{y}$ when the true class is y within the group g_k.

A key challenge in this framework is to design approaches achieving the most *useful partition* for predicting decision costs expectation. In the first article which presents this framework Dachraoui et al. (2015), a method, called ECONOMY-K, is designed as follows. (i) A partition of training examples is first performed by a K-means algorithm ; (ii) then a simple model uses the Euclidean distance as a proxy of the probability that $\mathbf{x}_t$ belongs to each group; (iii) the continuation of training time series within each group is exploited to predict the cost expectation for future time steps.

In order to avoid the clustering step with the associated choice of hyperparameters (Tavenard & Malinowski, 2016) presented a variant called NOCLUSTER which uses the 1-nearest neighbor in the training set in order to guess the likely future of $\mathbf{x}_t$.

Then, ECONOMY-γ was introduced by Achenchabe et al. (2021a) which relies on a supervised method to define a confidence-based partition of training time series. The algorithm, dedicated to binary classification problems, is designed as follows: (i) a collection of partitions is constructed by discretizing the output of each classifier $\{h_i\}_{i\in[1,T]}$ into equal-frequency intervals, the groups thus formed correspond to confidence levels for each time step; (ii) at the current time t, the incoming time series $\mathbf{x}_t$ belongs to only one group, since the output of the classifier h_t falls within a particular confidence level; (iii) then, a Markov chain model is trained to estimate the probabilities of the future time step confidence levels. ECONOMY-γ-MAX (Zafar et al., 2021) generalizes this approach to multi-class problems, aggregating the multiple conditional probabilities in the classifiers' output by using only the most probable class value.

• The BENEFITTER algorithm (Shekhar et al., 2023) is an anticipation-based approach that learns to predict the *benefit* of triggering a prediction early. This quantity is equal to the saving one could make making some decision now minus the cost induced by a wrong prediction. A LSTM model is learned to regress the benefit, which thus triggers, at inference time, as soon as the benefit is positive, i.e. when savings induced by temporal cost exceed estimated misclassification costs.

• CALIMERA (Bilski & Jastrzebska, 2023) uses anticipation about the future from another perspective. Instead of trying to guess the likely continuation of $\mathbf{x}_t$ which allows one to compute expected future costs, and therefore to wait until there seems no better time to make a prediction, their method is based on predicting directly the difference in cost between predicting the class now or waiting at least one more time step. If this difference is positive, then it is better to postpone the prediction. They advocate furthermore, that a calibration step should intervene above the regression in order to make a decision.

3.3 Reinforcement Learning based methods

To learn an ECTS function, it is possible to use an agent that learns what to do by exploring the outcomes associated with different decision strategies as it interacts with the world. The ECTS problem can therefore be recast as a Reinforcement Learning (RL) problem.

In RL, an agent must learn to associate an action a with each observable state s so that the expected gain is maximized. Let us suppose that the agent-environment interactions break naturally into subsequences which we call *episodes.*

At each time step t, the agent perceives the environment's state s_t (i.e. $\mathbf{x}_t$), chooses an action a_t (i.e. either make a prediction now and measure the gain $G_t = -\mathrm{C}_m(\hat{y}|y) - \mathrm{C}_d(\hat{t})$, or postpone the decision and receive the next measurement x_{t+1}) according to its current policy π, where $\pi(a|s) = p(a_t = a|s_t = s)$ is the probability that action a is taken given the state s.

The goal is for the agent to learn an optimal policy $\pi^\star$ from sequences of interactions with its environment $\langle s_t, a_t, s_{t+1}\rangle$. This can be done by computing the utility function $Q(s, a)$ defined for all (state, action) pairs.

By definition:

$$Q_\pi(s, a) \;=\; \mathbb{E}_\pi \left[G_t | S_t = s, A_t = a\right] \tag{12}$$

and the optimal policy can be derived from:

$$Q^\star(s, a) \;=\; \max_\pi Q_\pi(s, a) \tag{13}$$

It suffices at each observed state s to choose action $a^\star$ such that:

$$a^\star \;=\; \arg\max_a Q^\star(s,a) \tag{14}$$

One way to learn the function $Q^\star$ is using the Q-learning algorithm (Dayan & Watkins, 1992; Sutton & Barto, 2018)[5] and its variants for continuous definition of environment's states, such as Deep Q-learning (Mnih et al., 2015).

In *End-to-end* approaches, only one function is responsible for both stopping and making a prediction about the class of $\mathbf{x}_t$, whereas, in *Separable* approaches, two combined functions are involved, respectively dedicated to triggering the prediction and to the classification itself. In the case of Reinforcement Learning-based approaches, this distinction takes the following form:

1. *The Separable approach.* RL can be used to learn the *trigger* function only, once the classifiers h_t have been learned. In that case, the set of actions $\mathcal{A}_t$ at each time step t is restricted to two elements: {'*decide now*', '*postpone decision*'}. The Q function evaluates for each state the expected gain for each of the two possibilities, allowing one to decide what to do at each time step.

2. *The End-to-end approach.* RL can also be used to learn at once both when to trigger a prediction and what prediction to make. In principle, it suffices to extend the set of actions to $\mathcal{A}_t = \{\text{'}\textit{postpone decision}\text{'}, c_1, \ldots, c_N\}$, where there are N classes. Either the agent postpones the decision, or it predicts a class for the incoming time series $\mathbf{x}_t$.

In the literature, Reinforcement Learning-based ECTS approaches frequently include Deep Learning modules:

Separable RL approaches: The EARLIEST (Early and Adaptive Recurrent Label ESTimator) uses a RNN architecture (Hartvigsen et al., 2019) to make the prediction and a Reinforcement Learning agent trained jointly using policy gradient to trigger prediction or not. If a prediction is triggered, the hidden representation given by the RNN is sent to a Discriminator, whose role is to predict a class, given this representation. The model has been adapted to deal with irregularly sampled time series (Hartvigsen et al., 2022). Pantiskas et al. (2023) extend the ECTS framework to channel filtering, using here also Reinforcement Learning.

End-to-end RL approaches: Martinez et al. (2018; 2020) use a Deep Q-Network (Mnih et al., 2015), alongside a specifically designed reward signal, encouraging the agent to find a good trade-off between earliness and accuracy. Those types of approaches also naturally extend to online settings where time series are not of fixed length. Wang et al. (2024) introduce EarlyStop-RL, in which model-free RL is used to address the problem of early diagnosis of lung cancer.

3.4 Deep Learning based methods

Alongside the RL-based approaches, there exist deep learning methods that do not use RL.

The Decouple ETSC Network (DETSCNET) (Chen et al., 2022) architecture leverages a gradient projection technique in order to jointly learn two sub-modules: one for variable-length series classification, and the other for the early exiting task.

The End-to-end Learned Early Classification of Time Series method (ELECTS) leverages an LSTM architecture, adding a stopping prediction head to the network and adapting the loss function to promote good early predictions (Rußwurm et al., 2023).

[5] Note that other approaches are possible in the reinforcement learning scenario, like learning the utility function $V(s)$ and using TD-learning, or even learning the policy π directly. The Q-learning approach is however widely used.

4 Experiments & Results

This section presents the extensive set of experiments carried out in order to provide a consistent and fair evaluation of a wide range of the existing literature's methods. We first describe the experimental protocol used. We then turn to the experiments and their results. Figure 4 provides a synthetic view of the organization of these experiments.

- Section 4.1 introduces the experimental protocol as well as the global evaluation methodologies.
- In Section 4.2, the main state-of-the-art and the three baseline methods are evaluated using a widely used cost setting, i.e. with a a binary balanced misclassification cost and a linear delay cost.
- In Section 4.3, methods are tested in an anomaly detection scenario where the misclassification cost matrix is severely imbalanced, with false negatives being much more costly than false positives, and where the delay cost is no longer linear with time but increases exponentially with time.
- Finally, Section 4.4 briefly describes a set of other experimental setups, derived from either standard setting or the anomaly detection one, including for instance testing the impact of z-normalization. Complementary results can be found in Appendix C.

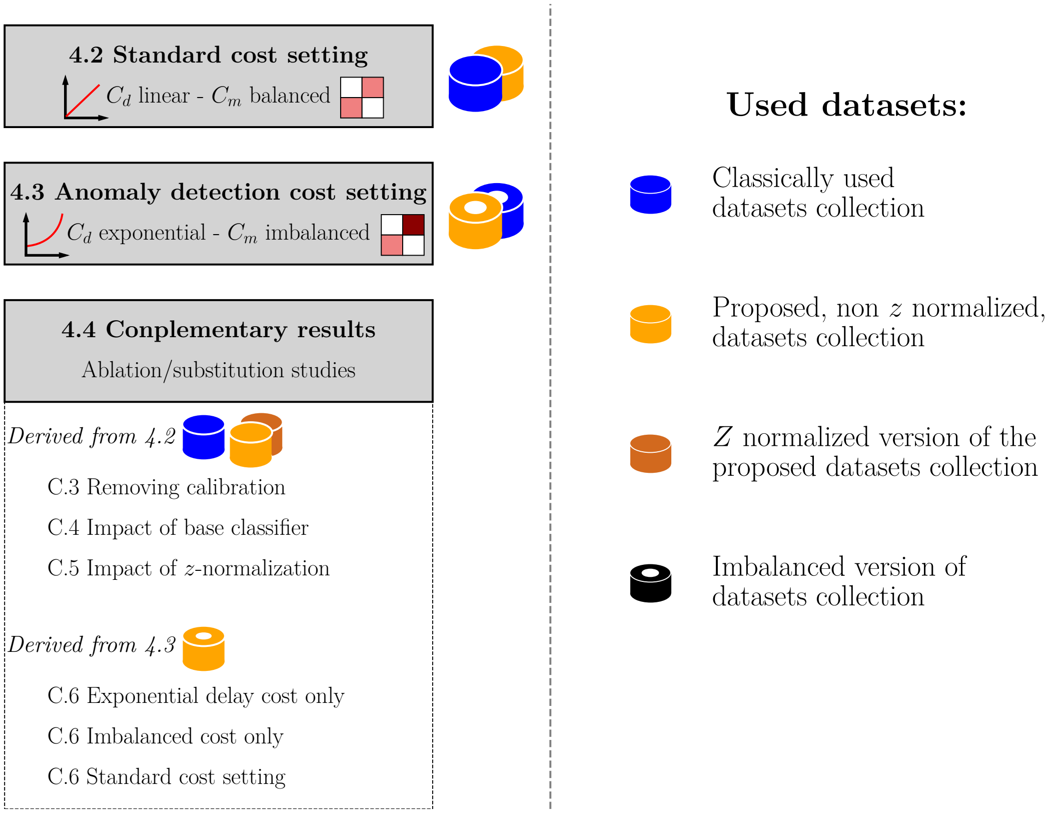

4.2 Standard cost setting
C_d linear - C_m balanced
4.3 Anomaly detection cost setting
C_d exponential - C_m imbalanced
4.4 Conplementary results
Ablation/substitution studies
Derived from 4.2
C.3 Removing calibration
C.4 Impact of base classifier
C.5 Impact of z-normalization
Derived from 4.3
C.6 Exponential delay cost only
C.6 Imbalanced cost only
C.6 Standard cost setting
Used datasets:
Classically used datasets collection
Proposed, non z normalized, datasets collection
Z normalized version of the proposed datasets collection
Imbalanced version of datasets collection

Figure 2: Experiments diagram: each line corresponds to one (or two) full benchmark runs, representing in total twelve full benchmarks. While this Section mainly discusses results about the cost settings in Subsections 4.2 and 4.3, many other alternative experiments are briefly analyzed in Subsection 4.4 and are more detailed in the Appendix C.

The experiments presented here aim to evaluate the effects of design choices on method performance and thus to provide answers to the questions:

- Do anticipation-based methods perform better than blind or myopic ones?
- Do methods that are cost-informed for their decision (i.e. explicitly estimating costs) perform better than methods that are cost-uninformed (e.g. confidence-based) (see Section 2.3.2).
- How the various methods fare when modifying the form of the delay cost and/or the misclassification cost matrix?

4.1 Experimental Protocol

This section covers the shared part of the experimental protocol for all experiments irrespective of the choice of the cost functions (see Sections 4.2 and 4.3 for this).

4.1.1 Evaluation of the performance

The ECTS problem explicitly balances the pressure to make a correct prediction and a time pressure. The correctness of the prediction is measured by the misclassification cost $\mathrm{C}_m(\hat{y}|y)$ where $\hat{y}$ is the prediction and y is the true class. The time pressure is sanctioned by a delay cost $\mathrm{C}_d(t)$ that is assumed to be positive and, in most applications, an increasing function of time. Sometimes, the two can be combined when the misclassification cost is a function of the time : $\mathrm{C}_{md}(\hat{y}|y,t)$.

Therefore, for each test time series $\mathbf{x}^i$, an ECTS method incurs a cost assumed to be of the following additive form: $\mathrm{C}_m(\hat{y}_i|y_i) + \mathrm{C}_d(\hat{t})$, where $\hat{t}$ is the time when the system decided to make a prediction, this prediction being $\hat{y}_i$.

For a test set of M time series, the *average cost* of a method is:

$$AvgCost_{\text{test}} \;=\; \frac{1}{M} \sum_{i=1}^{M} C_m(\hat{y}_i|y_i) + C_d(\hat{t}_i) \tag{15}$$

This is accordingly the criterion with which we *evaluate* the methods in the experiments reported in this paper.

In addition, in order to assess how the methods adapt to various balances between the misclassification and the delay costs, we vary the settings of these costs by weighting them during training and testing. The performance of the methods is therefore evaluated using the weighted average cost, as defined in Equation 16, for different values of the costs balance α, ranging from 0 to 1, with a 0.1 step:

$$AvgCost_{\alpha} = \frac{1}{M} \sum_{i=0}^{M} \alpha \times C_m(\hat{y}_i|y_i) + (1-\alpha) \times C_d(\hat{t}_i) \tag{16}$$

Small values of α correspond to a high delay cost and a small misclassification cost ; inversely, *large values* of α give more weight to the misclassification cost with a lower delay cost.

4.1.2 Optimization of the parameters of the methods

Approaches from the literature have been tested, respecting as far as possible the choices made in the original papers. The methods tested have two groups of hyperparameters: (*i*) some of them are meta parameters independent of the dataset and have been fixed according to the original papers, (*ii*) others have to be optimized using a grid search based on the *AvgCost* criterion. The optimization of the second group of hyperparameters has been realized using the value bounds according to the original published papers. When possible, the granularity of the grid has been adapted to keep similar computation times between competitors. These two groups of hyperparameters are described for all methods in the Appendix A. As a remark, because the original version of Teaser uses the Harmonic Mean (see Section 2.2), we have kept this setting (the resulting method being Teaser$_{HM}$), and we have added a variant called Teaser$_{Avg}$ optimized using *AvgCost*.

4.1.3 Comparing the trigger methods

In order to carry out a fair comparison between the tested methods, we isolated as far as possible the *triggering* function, responsible for deciding when to stop receiving measurements, from the *prediction* one, responsible for predicting a label to the incoming time series. As advocated by Bilski & Jastrzebska (2023), we have chosen the MiniROCKET algorithm (Dempster et al., 2021) to be the base classifier for all methods. It is indeed recognized as among the best performing classifiers in the time series classification literature as well as one of the fastest ones. Of course, distinguishing the decision component from the prediction one is only possible for the *separable* methods. In our experiments, we chose not to evaluate *end-to-end* methods, leaving that for future work.

Trigger models: Nine trigger models were selected from the literature based on their usage and their performances[6].

- Economy-γ-Max (Achenchabe et al., 2021a): triggers a decision if the predicted cost expectation is the lowest at time t when compared with the expected cost for all future time steps (cf. Section 3.2, Anticipation-based).
- Calimera (Bilski & Jastrzebska, 2023): triggers a decision when a regressor model which predicts the difference between the current observed cost and the minimum cost in the future is negative (cf. Section 3.2, Anticipation-based).
- Stopping Rule (Mori et al., 2017a): uses a trigger function based on a linear combination of confidence estimates and a delay measure linear on time (cf. Section 3.1, Confidence-based).
- Teaser$_{HM}$ (Schäfer & Leser, 2020): employs a trigger module consisting of a collection of T One Class SVM learned over the training set in order to isolate good predictions from bad ones. A prediction is triggered once ν consecutive predictions have been classified as 'good' by these OneClass SVM (ν being tuned to maximize the harmonic mean between *Earliness* and *Accuracy*) (cf. Section 3.1, Confidence-based).
- Teaser$_{Avg}$ (Schäfer & Leser, 2020): same algorithm as above. ν is now tuned maximizing the *AvgCost* criterion, in order to allow the method to adapt to different cost settings.
- Ecec (Lv et al., 2019): defines a confidence measure, based on the aggregated confidence of the predictions up to time t, and triggers a prediction if it exceeds a threshold, tuned by explicit grid-search (cf. Section 3.1, Confidence-based).
- Ecdire (Mori et al., 2017b): determines "safe" timestamps, based on classifier performance, from which predictions about possible classes can be made. Predictions cannot be triggered if those timestamps have not been reached. In addition, the difference between the two highest predicted probabilities must also exceed a certain threshold. (cf. Section 3.1, Confidence-based).
- Ects (He et al., 2013): computes the first time t for which nearest neighbors of the incoming time series $\mathbf{x}_t$ in the training set were given a label that did not change by the classifier (cf. Section 3.2, Anticipation-based).

All these methods have been re-implemented using Python, reproducing results close to the published ones. Except the code for the Ects implementation, which has been taken from Kladis et al. (2021). Hyperparameters are the ones chosen in the original published methods. Code to reproduce the experiments is available publicly at (see attached zip file).

Baselines: Furthermore, in order to evaluate the benefits, if any, of the various methods, it is telltale to compare them with simple ones. We chose three such baselines:

[6] The *EDSC* algorithm (Xing et al., 2011), even though available in the provided library, is not included in the following experiments, due to high space and time complexity (which hinders fair comparisons)

- Asap (As Soon As Possible) always triggers a prediction at the first possible timestep.
- Alap (As Late As Possible) always waits the complete series to trigger the prediction.
- Proba Threshold is a natural, confidence-based, cost-informed at training time, baseline: it triggers a prediction if the estimated probability of the likeliest prediction exceeds some threshold, found by grid search (cf. Section 3.1, Confidence-based).

4.1.4 Calibration of the classifications

Like Bilski & Jastrzebska (2023), we add a calibration step when learning the classifiers, i.e. Platt's scaling (Platt et al., 1999). Indeed, as we are dealing with collections of independently trained classifiers, the prediction scores may not remain consistent with one another over the time dimension. However, the trigger methods usually have their parameters set with the same values for all time steps. This is the case, for example with the Proba Threshold approach. In addition, some approaches such as Calimera and Economy-γ-Max exploit the estimated posterior probabilities $\{p(y|\mathbf{x}_t)\}_{y \in \mathcal{Y}}$ to estimate the future cost expectation. It is therefore highly desirable for all classifiers, at all times, to have their output calibrated and is necessary for a fair comparison.

4.1.5 Datasets and training protocol

Two collections of datasets[7]**:** In order to be able to directly compare our results to past experiments, we first use the usual TSC datasets from the UCR Archive (Dau et al., 2019) with the default split. In total, we have used 77 datasets from the UCR Archive, i.e. the ones with enough training samples to satisfy our experimental protocol end-to-end (blue cylinder in Figure 4). In this way, most of the datasets used by either Mori et al. (2017a) and Lv et al. (2019) or by Achenchabe et al. (2021a) are contained in our experiments.

A second collection of non z-normalized data sets is also provided. In this way, the associated potential information leakage is avoided (see Section 1). Any difference in the performance obtained on the z-normalized data sets can thus signal the danger of z-normalization with firm evidence. Considering the limited amount of non z-normalized datasets within the UCR archive (Dau et al., 2019), we have decided to look for complementary new datasets so as to provide another collection of datasets. To this end, the Monash archive for extrinsic regression (Tan et al., 2020), provided 20 new time series datasets, for which we have discretized the numeric target variable into binary classes based on a threshold value. For instance, if this threshold is equal to the median value of the regression target, the resulting classification datasets will be balanced in terms of classes (as in Section 4.2). Note that this threshold can be chosen differently to get imbalanced datasets (as in Section 4.3.2), several thresholds could also be used to increase the number of classes. As a result, we get a new set of classification tasks, as has recently been done by Middlehurst et al. (2024b). Finally, 34 datasets have been gathered: 14 from the original archive and 20 from the Monash extrinsic regression archive. (orange cylinder in Figure 4).

Datasets selection: An efficient ECTS approach is one that is capable of discriminating as soon as possible between incoming time series for which the decision is risky, or on the contrary certain, and adapting the trigger time accordingly. To properly evaluate these approaches, and to apply them to real-life applications, it is necessary to examine data and classifier performance over time. Let's consider two extreme cases: if class predictions are perfect (or even random) for all time steps $t \in [1, T]$, the optimal behavior will be to trigger decisions at the first time step for all series. Another undesirable case is when classification performance shows high variance over time, meaning there is still not enough information for predicting the target using the available classifier. This phenomenon can be due to a number of reasons, including: a limited training sample size, an unsuitable sampling frequency or noisy data. In this case, trigger model training is largely compromised, as it becomes impossible to adapt triggering time based on the classifier's behavior. Under these pathological cases, the application of ECTS methods makes no sense, and it would not be possible to properly evaluate competing approaches, since the strategy of triggering at the first time step would be close to the optimum. On the contrary, in the proposed benchmark, none of the 34 selected

[7] All original datasets of the paper can be downloaded, already prepared and splitted, from `https://urlz.fr/qRqu`

datasets fall into the pathological cases described above and they have been selected even more restrictively (see Appendix B), according to the following criteria: (i) the absence of z-normalization, (ii) an increasing trend in classification performance is observed over time, showing an information gain. We consider that satisfying these conditions is sufficient to reasonably choose a dataset on which to apply ECTS approaches, reflecting the existence of early clues for target prediction and a learnable classification task. As an intuitive example, Figure 3 displays the time series of a synthetic dataset, along with classification performance over time. In view of the work required to build up a corpus of databases suitable for ECTS and meeting the two conditions described above, we propose that this corpus, developed and presented in this document, be used by the community for future performance studies.

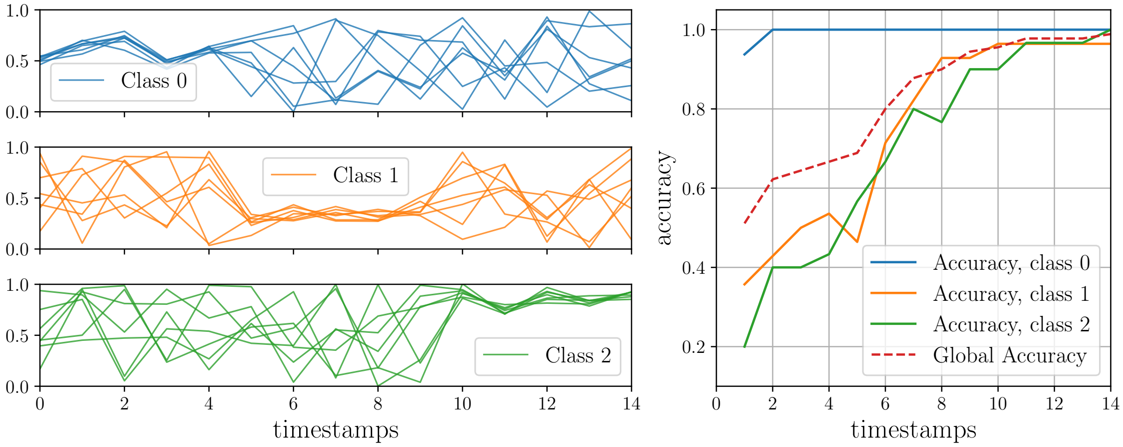


Figure 3: Examples of training time series of the *SmoothSubspace* dataset. Each of the class has a discriminative signal within one third of the serie.
Classification performances of a collection of classifiers, defined as a Spline transform followed by a linear Ridge model, calibrated using Platt's scaling.

Splitting strategy: When not using predefined splits, the train sets are split into two distinct sets in a stratified fashion: a first one to train the different classifiers, corresponding to 40% of the training set and another one to train the trigger model, trained over the 60% left. The set used to train the classifiers is itself split into two different sets in order to train calibrators, using 30% of the given data. Because of this procedure, we have been led to exclude some of the datasets, due to their limited training set size.

All the experiments have been performed using a linux operating machine, with an Intel Xeon E5-2650 2.20GHz (24-cores) and 252GB of RAM. Proceeding all datasets (including both blue and orange cylinders) over all competing approaches takes between 9-10 days, using MiniROCKET classifier, which is the most efficient tested.

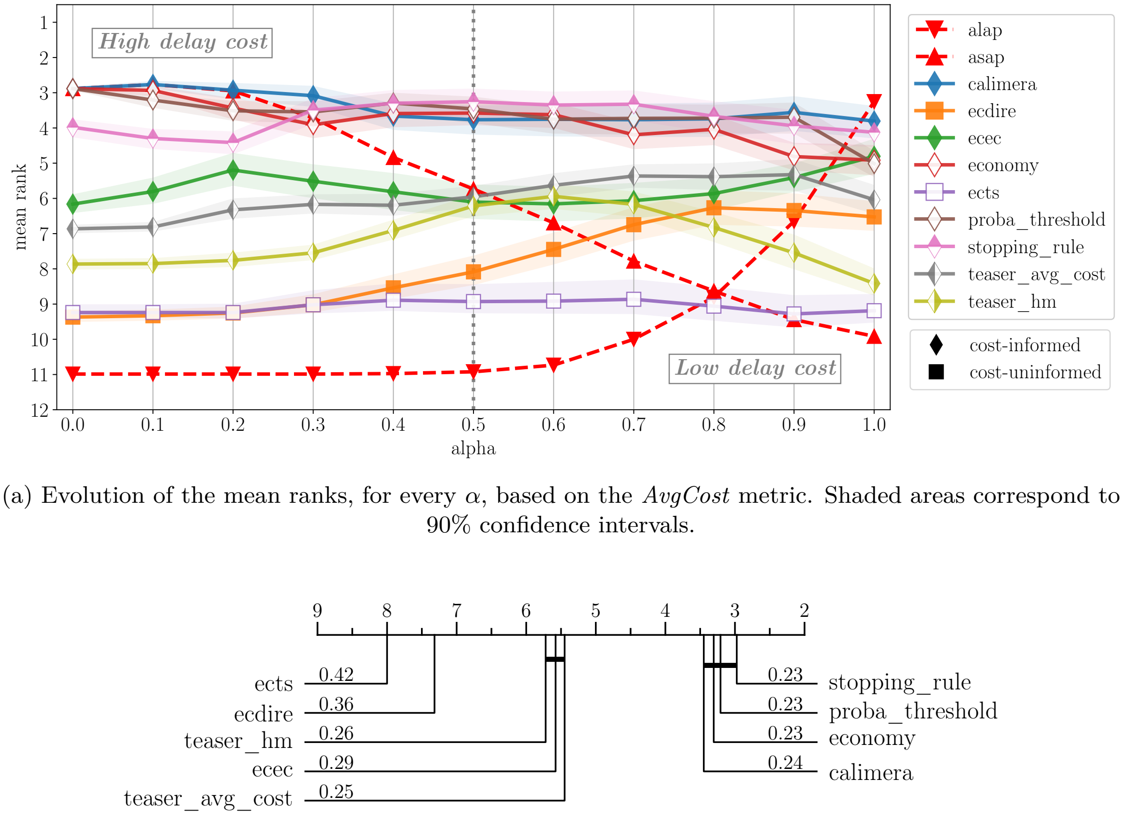
4.2 Experiments with balanced misclassification and linear delay costs

This first setting is the one most widely used in the literature to date.

4.2.1 Cost definition

The misclassification cost is symmetrical and balanced; the delay cost is linear. They can be defined as follows:

$$C_m(\hat{y}|y) = \mathbb{1}(\hat{y} \neq y)$$
$$C_d(t) = \frac{t}{T}$$



(a) Evolution of the mean ranks, for every α, based on the *AvgCost* metric. Shaded areas correspond to 90% confidence intervals.

(b) Alpha is now fixed to $\alpha = 0.5$. Wilcoxon signed-rank test labeled with mean *AvgCost*.

Figure 4: The ranking plot (a) shows that, across all values of α, a top group of four approaches distinguishes itself. The significance of this result is supported by statistical tests. Specifically, we report this for $\alpha = 0.5$ as shown in (b).

Thus, for each dataset, the $AvgCost_\alpha$ is bounded between 0 and 1.

4.2.2 Results and analysis

For comparability reasons, this first set of experiments is analyzed over the classical ECTS benchmark used in the literature so far (blue cylinder in Figure 4). Results over the new, non z-normalized, datasets can be found in Appendix C.

Figure 4b provides a global view about the relative performances of the tested methods. The Wilcoxon-Holm Ranked test provides an overall statistical analysis. It examines the critical difference among all techniques to plot the method's average rank in a horizontal bar. Lower ranks denote better performance, and the methods connected by a horizontal bar are similar in terms of statistical significance. When evaluated by their *average rank* on all data sets with respect to the average cost (Equation 16), here for $\alpha = 0.5$, four methods significantly outperform the others:

Methods	Confidence	Anticipation	Cost informed
Stopping Rule	✓		train
Proba Threshold	✓		train
Economy-γ-max		✓	train & test
Calimera		✓	train

Figure 4a allows a closer look, this time varying the relative costs of misclassification and delaying prediction using Equation 16, where a small value of α means that delay cost is paramount. 90% level confidence intervals have been computed using bootstrap[8]. Again, the same four methods top the others for almost every values of α. Not surprisingly, the baseline Asap (predict as soon as possible) is very good when the delay cost is very high, while Alap (predict at time T) is very good when there is no cost associated with delaying decision.

It is remarkable that, in this cost setting, the simple Proba Threshold method exhibits a strong performance for almost all values of α. It is therefore worth including in the evaluation of new methods. However, while Figures 4a, 4b are useful for general analysis, they do not provide insights about how the Accuracy vs. Earliness trade-off is optimized for each of the competitors. Figure 5 provides some explanation for this.

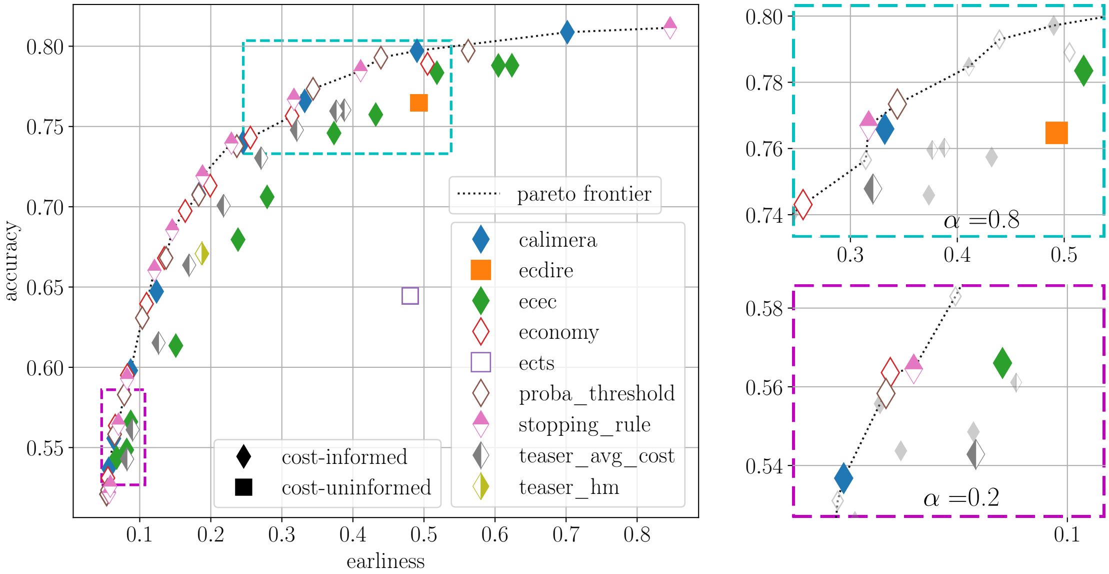


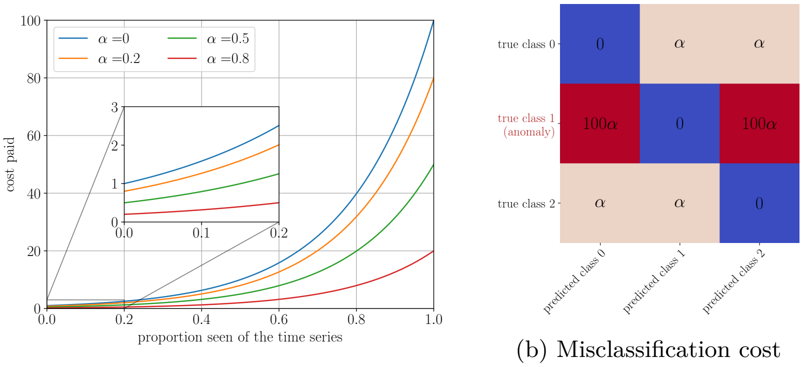
Figure 5: Pareto front, displaying for each α the *Accuracy* on the y axis and *Earliness* on the x axis. Best approaches are located on the top left corner. In zoomed boxes, on the right of the Figure, points corresponding to a single α are highlighted, while other points are smaller and gray. Each of the trigger model is optimizing the trade-off in its own way, resulting in many different approaches having points in the Pareto dominant set.

In this figure, the two evaluation measures: '*Accuracy*' and '*Earliness*', are considered as dimensions in conflict. The *Pareto front* is the set of points for which no other point dominates with respect to both *Accuracy* and *Earliness*. It is drawn here when varying their relative importance using α (in the set $\{0, 0.1, 0.2, \ldots, 1.0\}$).

One must note first that, as Ects and Ecdire are cost-uninformed, their performance does not vary with α. Whatever the relative weight between accuracy and earliness, they make their prediction approximately after having observed half of the time serie and they reach an average accuracy respectively near 0.64 and 0.77. They are clearly dominated by the other methods. This is also the case for Teaser$_{HM}$, which, while being *cost-informed at training time*, also only appears once in the figure. Indeed, no weighting mechanism is provided in the original version of the algorithm, where the harmonic mean is used as an optimization criterion (see Equation 5).

Each of the leading methods Stopping Rule, Proba Threshold, Economy and Calimera have at least one point on the Pareto front and generally exhibits a combined performance very close to it. A closer look

[8] Resample with replacement has been done a large number of times (10.000×) and are reported as shaded colors on the figure. The statistic of interest is studied, here the mean, by examining the bootstrap distribution at the desired confidence level.



(a) Exponential delay cost: $C_d(t) = (1 - \alpha)\exp(\frac{t}{T} * \log(100))$

(b) Misclassification cost matrix (three class problem)

Figure 6: Representative delay cost (a) and misclassification ones (b) for an anomaly detection scenario. In our experiments, $\alpha \in [0, 1]$.

reveals how each approach optimizes the *earliness* vs. *accuracy* trade-off differently for a fixed cost. If we consider $\alpha = 0.8$, for example, it appears that Economy takes its decision earlier than Proba Threshold, itself being more precocious than Ecec. Because this is also an area of problems where the delay cost is low, by doing so, Economy prevents itself from benefiting from waiting for more measurements and increasing its performance. Hence its slight downward slope on Figure 4a for high values of α.

It is worth noting that the two naive baselines Asap and Alap perform better than the majority of approaches on seven α values out of ten. This is especially the case when the delay cost is large, i.e. for $\alpha \in [0.1, 0.3]$, for which the Asap baseline is as competitive as top performers. Globally, the performance of Proba Threshold is remarkable in this cost setting. Even though it is simply based on a single threshold on the confidence in the current prediction, its performance makes it one of the best methods.

The results computed over the proposed datasets ensemble (i.e. orange cylinder) are displayed in Figure 9 of Appendix C. No significant changes can be observed in the ranking of competing approaches.

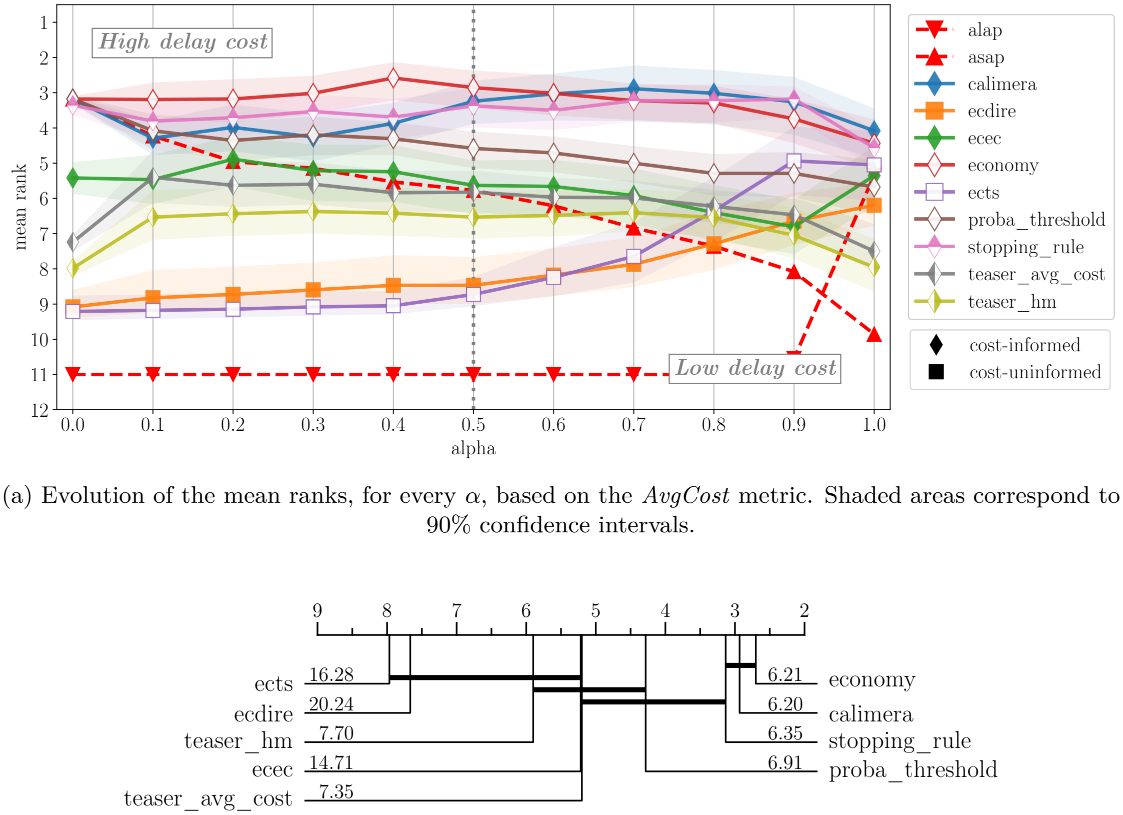
One question is how such a simple method, like Proba Threshold, can adapt to scenarios where the misclassification and delay costs, not being symmetrical for the misclassification cost, and not linear for the delay one, reflect other application settings.

4.3 Experiments with *unbalanced* misclassification and *non-linear* delay costs

While the previous section has provided a first assessment of how the various methods adapt to different respective weights for the misclassification and the delay costs, it nonetheless assumed that the misclassification costs were balanced (e.g. 0 if correctly classified and 1 otherwise) and that the delay cost was a linear function of time.

There are however applications where these assumptions do not hold, for instance predictive maintenance or hospital emergency services, are characterized by (i) *imbalanced* misclassification costs (e.g. it is more costly to have to repair a machine than to carry out a maintenance operation that turns out not being necessary) and by (ii) *non linear* delay cost (e.g. usually, the later the surgical operation is decided, the costlier it is to organize it and the larger the risk for the patient). In the following, we call all applications presenting these characteristics “anomaly detection” applications.

The question arises as to how the various ECTS algorithms behave in this case, depending on their level of cost awareness and whether or not they are anticipation-based. This is what is investigated in the series of experiments reported in this section.



(a) Evolution of the mean ranks, for every α, based on the *AvgCost* metric. Shaded areas correspond to 90% confidence intervals.

(b) Alpha is now fixed to $\alpha = 0.5$. Wilcoxon signed-rank test labeled with mean *AvgCost*.

Figure 7: The ranking plot (a) shows that, across all α, a top group composed by three approaches distinguish. This result is significant as supported by statistical tests. Specifically, for $\alpha = 0.5$ as shown in (b).

4.3.1 Cost definition for anomaly detection

In order to study the behavior of the various algorithms on scenarios corresponding to anomaly detection, we set the *unbalanced misclassification cost* matrix such that a false negative (i.e. missing an anomaly) was 100 times costlier than a false positive (i.e. wrongly predicting an anomaly) (see Figure 6b). For this last situation, the delay cost was arbitrarily set to 1. The *delay cost* is defined as an exponential function of time. In order to have a delay cost commensurable with the misclassification one, we decided that waiting for the entire time series to be seen, at T, would cost $100 \times \alpha$ (see Figure 6a), starting at $(1 - \alpha)$ for $t = 0$ and reaching $100 \times \alpha$ when $t = T$.

4.3.2 Results and analysis

In this part, as a new cost setting is explored, there is no need to produce comparable results from previous works. Thus, we choose to use the new non z-normalized datasets collection (orange cylinder in Figure 4). In order for the imbalanced misclassification cost to make sense, those datasets have been altered so that the minority class represents 20% of all labels. As explained in Section 4.1, some extrinsic regression datasets are turned into classification ones. In these cases, the threshold value has been set to the second decile of the regression target. For the original classification datasets, the minority class has been sub-sampled when necessary.

Results from the Wilcoxon-Holm Ranked test (both regarding the average rank and the value for *AvgCost*) (see Figure 7b) and from the AvgCost plot (see Figure 7a) with varying values of α (in Equation 16) show that now the best method overall is Economy which is both cost-informed at training and testing time, beside being anticipating-based. However, Stopping-Rule is a very strong contender while being cost-informed at training time but not at testing time and confidence-based. There is a reason for it. When Stopping Rule equals or overpasses Economy, this is for high values of α when the delay cost loses its importance, therefore leaving the misclassification cost to reign and confidence-based methods to be good.

It may come as a surprise that Calimera lags behind Economy for $\alpha \in [0, 0.4]$, despite being similarly based on the estimation of future cost expectations. One reason for this is that the cost expectation is achieved by considering only the predicted class. This poor estimate of the cost expectancy becomes critical when the delay cost is important.

Similarly, Proba Threshold is surprisingly good in this scenario, even if it is no longer in the top tier. Looking solely at prediction confidence, we might expect it to be blind to the rapid increase in delay cost in the anomaly detection scenario. However, it is noticeable that the cost of delay only increases sharply after around 60% of the complete time series has been observed, which is generally sufficient to exceed the confidence threshold. Hence, Proba Threshold does not suffer from high delay costs that are to come, and exhibits good performance here.

Figure 8 plots the Pareto front considering two axes based on decision costs. The horizontal axis corresponds to the average delay cost incurred for each example, normalized by the worst delay cost paid at $t = T$. It is better to be on the left of the x-axis. The vertical axis corresponds to 1 minus the misclassification cost incurred for each example, normalized by the worst prediction cost. It is better to be high on the y-axis.

We observe that the Pareto front is composed almost exclusively of points corresponding to the Economy method. This is consistent with the evaluation based on the *AvgCost* metric. This figure highlights the fact that the design of approaches capable of handling arbitrarily parameterized decision costs requires a cost-informed application framework.

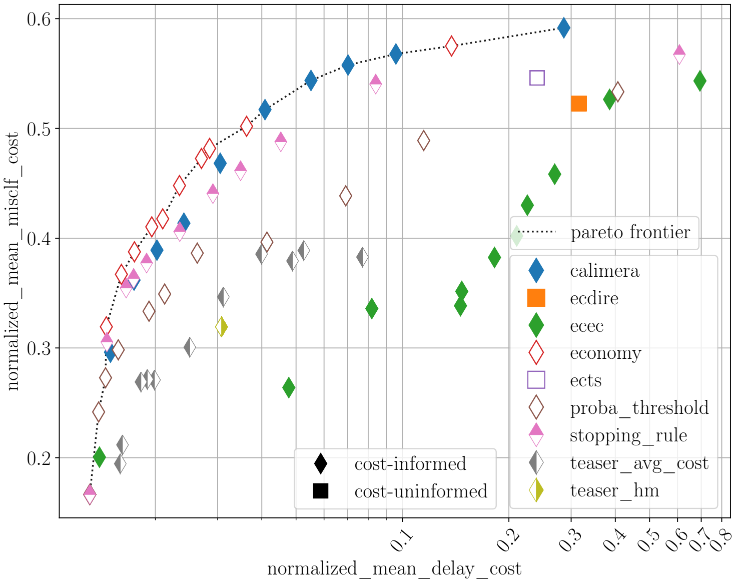


Figure 8: Pareto front, displaying for each α, the normalized version of the *AvgCost*, decomposed over delay and misclassification cost on x-axis and y-axis respectively. Best approaches are located on the top left corner. Due to the exponential shape of the delay cost, the x-axis is on log scale.

4.4 Other experiments: ablation and substitution studies

In this section, complementary experiments, namely ablation studies as well as sanity check are briefly discussed. For the sake of brevity, the figures supporting the analysis are reported in Appendix C.

Impact of removing calibration

Bilski & Jastrzebska (2023) assert that calibration of the classifiers is paramount for the performance of ECTS algorithms. In order to test this claim, we have repeated the experiment removing the calibration step. The examples used for calibration have been removed as well during training, so that all else remains the same as before.

The results of Figure 12 show that indeed Calimera suffers greatly if no calibration is done. Indeed, this approach relies on estimating the expectation of future costs via a regression problem, and a miscalibration may have a negative impact on the built targets. For its part, Proba Threshold suffers somewhat mildly. This is no surprise as they rely on a single threshold on the confidence of the prediction for all time steps.

Impact of the choice of base classifier

All methods have been compared using the same classifier: MiniROCKET so that only the decision components differ. However, the choice of the base classifiers could induce a bias favoring or hampering some methods. In order to clarify this, we have repeated the experiments replacing MiniROCKET with two base classifiers: WEASEL 2.0 (Schäfer & Leser, 2023), and the XGBoost classifier (Chen et al., 2015) using features produced by tsfresh (Christ et al., 2018). Both of these classifiers have already been tested within the ECTS literature by Schäfer & Leser (2020); Lv et al. (2019) and Achenchabe et al. (2021a) respectively. Figure 13 and 14 in Appendix C.4 report the results respectively with these two classification methods. One can observe that the results are not significantly altered with the same overall ordering of the methods when varying the value of α. Furthermore, our results on *AvgCost* show that performance tends to be better for all methods using MiniROCKET (see Table 5 in Appendix C.4). It is thus to be preferred given its simplicity and good performance.

Impact of z-normalization

Considering the newly proposed ensemble of datasets, we were not able to identify any problems of information leakage over time. This inconclusive result simply indicates that the variance of the time series measurements is not informative for these datasets, which still could be the case considering past published results. For further details, please refer to Section C.5 of Appendix C.

5 Conclusion

The first contribution of this research work is the coding of all the methods tested and making the codes available in a repository open to everyone. In this way, the experiments reported can be duplicated and further studies carried out. Furthermore, the deposited datasets and the experimental framework provide a ground for fair comparisons between competing methods. We claim that the *AvgCost* is the appropriate measure by which to evaluate the performance of the methods. This is indeed what will be "paid" at the end of the day by a practitioner using a method. We have accordingly characterized a number of methods from the literature.

Our extensive experiments have shown that:

- It is worthwhile to resort to dedicated ECTS methods, and more so in scenarios like anomaly detection with asymmetrical misclassification cost and exponential delay cost.
- However, it is noticeable that Proba Threshold, a baseline method, is surprisingly good overall in the standard setting with symmetrical misclassification cost and linear delay cost, exhibiting comparable performance to confidence-based myopic methods such as Stopping-Rule and anticipation-based cost-informed ones such as Calimera and Economy.
- Calibration of the classifiers has a large impact on some methods such as Calimera in particular, less so on other methods like Proba Threshold and Ecdire.

In this paper, we have proposed a reading guide to highlight the main characteristics of ECTS methods, namely (*i*) the importance of the two components: decision and prediction which are distinct in the "separable" architecture and not in the "end-to-end" one, (*ii*) the distinction between anticipation-based and myopic methods, and (*iii*) between cost-informed and cost-uninformed techniques. On the basis of these dimensions, it becomes easy to imagine new methods that combine them in original ways, which could lead to new properties and better performance for solving the problem of early classification of time series, which, present in many applications, has potentially great impacts.

To go a step further, future work could be carried out to study the literature's approaches applied in as yet unexplored cost settings. For example, in many applications, the delay cost depends on the true class and the predicted one, and thus a single cost function integrating misclassification and delay costs should then be used. The use of this general cost form requires the adaptation of some state-of-the-art methods and has not yet been studied.

In addition, in real ECTS applications, it is up to the business expert to define the costs, which is not an easy task in practice. Among the challenges, applications where the costs actually paid are not deterministic are of key interest (e.g. a manufacturing defect on an engine part does not necessarily lead to a failure, but it does increase the probability of paying a higher cost). Thus, future work could study the impact of *stochastic cost functions*. Another interesting case is applications where the costs paid are slightly different, or changed, from those defined by the business experts for the training phase (e.g. a change in the price of raw materials). Those kinds of *cost drift* between training and testing stages could also be further studied.

Finally, in the case of existing separable approaches, the misclassification cost is not exploited for training the classification function. Future work could investigate the interest of using cost-sensitive classifiers in the case of ECTS.

References

Youssef Achenchabe, Alexis Bondu, Antoine Cornuéjols, and Asma Dachraoui. Early classification of time series: Cost-based optimization criterion and algorithms. Machine Learning, 110(6):1481–1504, 2021a.

Youssef Achenchabe, Alexis Bondu, Antoine Cornuéjols, and Vincent Lemaire. Early classification of time series is meaningful. arXiv preprint arXiv:2104.13257, 2021b.

Carlos J Alonso González and Juan J Rodríguez Diez. Boosting interval-based literals: Variable length and early classification. In Data mining in time series databases, pp. 149–171. World Scientific, 2004.

Anastasios N Angelopoulos, Stephen Bates, Emmanuel J Candès, Michael I Jordan, and Lihua Lei. Learn then test: Calibrating predictive algorithms to achieve risk control. arXiv preprint arXiv:2110.01052, 2021.

Alessandro Antonucci, Mauro Scanagatta, Denis Deratani Mauá, and Cassio Polpo de Campos. Early classification of time series by hidden markov models with set-valued parameters. In Proceedings of the NIPS time series workshop, pp. 1–5, 2015.

James O Berger. Statistical decision theory and Bayesian analysis. Springer Science & Business Media, 1985.

Jakub Michal Bilski and Agnieszka Jastrzebska. Calimera: A new early time series classification method. Information Processing & Management, 60(5):103465, 2023.

Alexis Bondu, Youssef Achenchabe, Albert Bifet, Fabrice Clérot, Antoine Cornuéjols, Joao Gama, Georges Hébrail, Vincent Lemaire, and Pierre-François Marteau. Open challenges for machine learning based early decision-making research. ACM SIGKDD Explorations Newsletter, 24(2):12–31, 2022.

Huiling Chen, Ye Zhang, Aosheng Tian, Yi Hou, Chao Ma, and Shilin Zhou. Decoupled early time series classification using varied-length feature augmentation and gradient projection technique. Entropy, 24 (10):1477, 2022.

Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, Kailong Chen, Rory Mitchell, Ignacio Cano, Tianyi Zhou, et al. Xgboost: extreme gradient boosting. R package version 0.4-2, 1(4):1–4, 2015.

Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W Kempa-Liehr. Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package). Neurocomputing, 307:72–77, 2018.

Asma Dachraoui, Alexis Bondu, and Antoine Cornuéjols. Early classification of time series as a non myopic sequential decision making problem. In Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015, Proceedings, Part I 15, pp. 433–447. Springer, 2015.

Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The ucr time series archive. IEEE/CAA Journal of Automatica Sinica, 6(6):1293–1305, 2019.

Peter Dayan and CJCH Watkins. Q-learning. Machine learning, 8(3):279–292, 1992.

Morris H DeGroot. Optimal statistical decisions. John Wiley & Sons, 2005.

Angus Dempster, Daniel F Schmidt, and Geoffrey I Webb. Minirocket: A very fast (almost) deterministic transform for time series classification. In Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining, pp. 248–257, 2021.

Akinori F Ebihara, Taiki Miyagawa, Kazuyuki Sakurai, and Hitoshi Imaoka. Toward asymptotic optimality: Sequential unsupervised regression of density ratio for early classification. In ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5. IEEE, 2023.

Akinori F Ebihara, Taiki Miyagawa, Kazuyuki Sakurai, and Hitoshi Imaoka. Learning the optimal stopping for early classification within finite horizons via sequential probability ratio test. In The Thirteenth International Conference on Learning Representations, 2025.

Thomas S Ferguson. Who solved the secretary problem? Statistical science, 4(3):282–289, 1989.

Mohamed F Ghalwash and Zoran Obradovic. Early classification of multivariate temporal observations by extraction of interpretable shapelets. BMC bioinformatics, 13:1–12, 2012.

Mohamed F Ghalwash, Vladan Radosavljevic, and Zoran Obradovic. 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, pp. 402–411, 2014.

Bhaskar Kumar Ghosh and Pranab Kumar Sen. Handbook of sequential analysis. CRC Press, 1991.

Ashish Gupta, Hari Prabhat Gupta, Bhaskar Biswas, and Tanima Dutta. Approaches and applications of early classification of time series: A review. IEEE Transactions on Artificial Intelligence, 1(1):47–61, 2020.

Thomas Hartvigsen, Cansu Sen, Xiangnan Kong, and Elke Rundensteiner. Adaptive-halting policy network for early classification. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 101–110, 2019.

Thomas Hartvigsen, Walter Gerych, Jidapa Thadajarassiri, Xiangnan Kong, and Elke Rundensteiner. Stop&hop: Early classification of irregular time series. In Proceedings of the 31st ACM International Conference on Information & Knowledge Management, pp. 696–705, 2022.

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

Guoliang He, Yong Duan, Tieyun Qian, and Xu Chen. Early prediction on imbalanced multivariate time series. In Proceedings of the 22nd ACM international conference on Information & Knowledge Management, pp. 1889–1892, 2013.

Guoliang He, Yong Duan, Rong Peng, Xiaoyuan Jing, Tieyun Qian, and Lingling Wang. Early classification on multivariate time series. Neurocomputing, 149:777–787, 2015.

Ali Ismail-Fawaz, Angus Dempster, Chang Wei Tan, Matthieu Herrmann, Lynn Miller, Daniel F Schmidt, Stefano Berretti, Jonathan Weber, Maxime Devanne, Germain Forestier, and Geoff I Webb. An approach to multiple comparison benchmark evaluations that is stable under manipulation of the comparate set. arXiv preprint arXiv:2305.11921, 2023.

Evgenios Kladis, Charilaos Akasiadis, Evangelos Michelioudakis, Elias Alevizos, and Alexandros Artikis. An empirical evaluation of early time-series classification algorithms. In EDBT/ICDT Workshops, 2021.

Achim Klenke. Probability theory: a comprehensive course. Springer Science & Business Media, 2013.

Yu-Feng Lin, Hsuan-Hsu Chen, Vincent S Tseng, and Jian Pei. Reliable early classification on multivariate time series with numerical and categorical attributes. In Advances in Knowledge Discovery and Data Mining: 19th Pacific-Asia Conference, PAKDD 2015, Ho Chi Minh City, Vietnam, May 19-22, 2015, Proceedings, Part I 19, pp. 199–211. Springer, 2015.

Junwei Lv, Xuegang Hu, Lei Li, and Peipei Li. An effective confidence-based early classification of time series. IEEE Access, 7:96113–96124, 2019.

Junwei Lv, Yuqi Chu, Jun Hu, Peipei Li, and Xuegang Hu. Second-order confidence network for early classification of time series. ACM Transactions on Intelligent Systems and Technology, 2023.

Coralie Martinez, Guillaume Perrin, Emmanuel Ramasso, and Michèle Rombaut. A deep reinforcement learning approach for early classification of time series. In 2018 26th European Signal Processing Conference (EUSIPCO), pp. 2030–2034. IEEE, 2018.

Coralie Martinez, Emmanuel Ramasso, Guillaume Perrin, and Michèle Rombaut. Adaptive early classification of temporal sequences using deep reinforcement learning. Knowledge-Based Systems, 190:105290, 2020.

Chirag Mathukia, WuQiang Fan, Karen Vadyak, Christine Biege, and Mahesh Krishnamurthy. Modified early warning system improves patient safety and clinical outcomes in an academic community hospital. Journal of community hospital internal medicine perspectives, 5(2):26716, 2015.

Matthew Middlehurst, Ali Ismail-Fawaz, Antoine Guillaume, Christopher Holder, David Guijo Rubio, Guzal Bulatova, Leonidas Tsaprounis, Lukasz Mentel, Martin Walter, Patrick Schäfer, et al. aeon: a python toolkit for learning from time series. arXiv preprint arXiv:2406.14231, 2024a.

Matthew Middlehurst, Patrick Schäfer, and Anthony Bagnall. Bake off redux: a review and experimental evaluation of recent time series classification algorithms. Data Mining and Knowledge Discovery, pp. 1–74, 2024b.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. nature, 518(7540):529–533, 2015.

Usue Mori, Alexander Mendiburu, Sanjoy Dasgupta, and Jose A Lozano. Early classification of time series by simultaneously optimizing the accuracy and earliness. IEEE transactions on neural networks and learning systems, 29(10):4569–4578, 2017a.

Usue Mori, Alexander Mendiburu, Eamonn Keogh, and Jose A Lozano. Reliable early classification of time series based on discriminating the classes over time. Data mining and knowledge discovery, 31:233–263, 2017b.

Leonardos Pantiskas, Kees Verstoep, Mark Hoogendoorn, and Henri Bal. Multivariate time series early classification across channel and time dimensions. arXiv preprint arXiv:2306.14606, 2023.

Nathan Parrish, Hyrum S Anderson, Maya R Gupta, and Dun Yu Hsiao. Classifying with confidence from incomplete information. The Journal of Machine Learning Research, 14(1):3561–3589, 2013.

John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Advances in large margin classifiers, 10(3):61–74, 1999.

Liran Ringel, Regev Cohen, Daniel Freedman, Michael Elad, and Yaniv Romano. Early time classification with accumulated accuracy gap control. arXiv preprint arXiv:2402.00857, 2024.

Marc Rußwurm, Nicolas Courty, Rémi Emonet, Sébastien Lefèvre, Devis Tuia, and Romain Tavenard. 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, 2023.

Azusa Sawada, Taiki Miyagawa, Akinori Ebihara, Shoji Yachida, and Toshinori Hosoi. Convolutional neural networks for time-dependent classification of variable-length time series. In 2022 International joint conference on neural networks (IJCNN), pp. 1–8. IEEE, 2022.

Patrick Schäfer and Ulf Leser. Teaser: early and accurate time series classification. Data mining and knowledge discovery, 34(5):1336–1362, 2020.

Patrick Schäfer and Ulf Leser. Weasel 2.0: a random dilated dictionary transform for fast, accurate and memory constrained time series classification. Machine Learning, 112(12):4763–4788, 2023.

Shubhranshu Shekhar, Dhivya Eswaran, Bryan Hooi, Jonathan Elmer, Christos Faloutsos, and Leman Akoglu. Benefit-aware early prediction of health outcomes on multivariate eeg time series. Journal of biomedical informatics, 139:104296, 2023.

Larry A Shepp. Explicit solutions to some problems of optimal stopping. The Annals of Mathematical Statistics, 40(3):993, 1969.

Sergii Skakun, Belen Franch, Eric Vermote, Jean-Claude Roger, Inbal Becker-Reshef, Christopher Justice, and Nataliia Kussul. Early season large-area winter crop mapping using modis ndvi data, growing degree days information and a gaussian mixture model. Remote Sensing of Environment, 195:244–258, 2017.

Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. MIT press, 2018.

Chang Wei Tan, Christoph Bergmeir, Francois Petitjean, and Geoffrey I Webb. Monash university, uea, ucr time series extrinsic regression archive. arXiv preprint arXiv:2006.10996, 2020.

Alexander Tartakovsky, Igor Nikiforov, and Michele Basseville. Sequential analysis: Hypothesis testing and changepoint detection. CRC press, 2014.

Romain Tavenard and Simon Malinowski. Cost-aware early classification of time series. In Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I 16, pp. 632–647. Springer, 2016.

Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. Neural networks, 22(5-6):544–557, 2009.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

Abraham Wald and Jacob Wolfowitz. Optimum character of the sequential probability ratio test. The Annals of Mathematical Statistics, pp. 326–339, 1948.

Wenlin Wang, Changyou Chen, Wenqi Wang, Piyush Rai, and Lawrence Carin. Earliness-aware deep convolutional networks for early time series classification. arXiv preprint arXiv:1611.04578, 2016.

Yifan Wang, Qining Zhang, Lei Ying, and Chuan Zhou. Deep reinforcement learning for early diagnosis of lung cancer. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, pp. 22410–22419, 2024.

Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In 2017 International joint conference on neural networks (IJCNN), pp. 1578–1585. IEEE, 2017.

Renjie Wu, Audrey Der, and Eamonn Keogh. When is early classification of time series meaningful. IEEE Transactions on Knowledge and Data Engineering, 2021.

Zhengzheng Xing, Jian Pei, Guozhu Dong, and Philip S Yu. Mining sequence classifiers for early prediction. In Proceedings of the 2008 SIAM international conference on data mining, pp. 644–655. SIAM, 2008.

Zhengzheng Xing, Jian Pei, and S Yu Philip. Early prediction on time series: A nearest neighbor approach. In IJCAI, pp. 1297–1302. Citeseer, 2009.

Zhengzheng Xing, Jian Pei, Philip S Yu, and Ke Wang. Extracting interpretable features for early classification on time series. In Proceedings of the 2011 SIAM international conference on data mining, pp. 247–258. SIAM, 2011.

Zhengzheng Xing, Jian Pei, and Philip S Yu. Early classification on time series. Knowledge and information systems, 31:105–127, 2012.

Wenhe Yan, Guiling Li, Zongda Wu, Senzhang Wang, and Philip S Yu. Extracting diverse-shapelets for early classification on time series. World Wide Web, 23:3055–3081, 2020.

Liuyi Yao, Yaliang Li, Yezheng Li, Hengtong Zhang, Mengdi Huai, Jing Gao, and Aidong Zhang. Dtec: Distance transformation based early time series classification. In Proceedings of the 2019 SIAM International Conference on Data Mining, pp. 486–494. SIAM, 2019.

Lexiang Ye and Eamonn Keogh. Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. Data mining and knowledge discovery, 22:149–182, 2011.

Paul-Emile Zafar, Youssef Achenchabe, Alexis Bondu, Antoine Cornuéjols, and Vincent Lemaire. Early classification of time series: Cost-based multiclass algorithms. In 2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA), pp. 1–10. IEEE, 2021.

Wenjing Zhang and Yuan Wan. Early classification of time series based on trend segmentation and optimization cost function. Applied Intelligence, pp. 1–12, 2022.

A Hyperparameters

Table 2: Hyperparameters' value. Δ defines grid's size when performing grid-search over continuous valued intervals.

	Hyperparameters	
Method	Fixed	Optimized
Calimera	*kernel*: "rbf"	
Ecdire	$\mathit{perc\_acc} = 100\%$	
Ects	$\mathit{support} = 0$	
Economy		$k \in [\![1 \,..\, 20]\!]$
Proba Threshold		$\Delta = 40$
Stopping Rule		$\gamma_1,\ \gamma_2,\ \gamma_3 \in [-1, 1]$ $\Delta = 10^3$
Teaser_*		$\nu \in [\![1 \,..\, 5]\!]$

B Data description

B.1 UCR Time Series Classification datasets

Table 3: UCR TSC datasets : 77 datasets from the UCR archive have been retained to run the experiments over the 128 contained in the full archive. Those are the ones with fixed length, without missing values and with enough training samples to execute our experiments pipeline end-to-end. *Italic* datasets are not included in experiments using default split for this reason.

Data	Train	Test	Length	Class	Type
ACSF1	100	100	1460	10	Device
Adiac	390	391	176	37	Image
Beef	30	30	470	5	Spectro
BeetleFly	20	20	512	2	Image
BME	30	150	128	3	Simulated
Car	60	60	577	4	Sensor
CBF	30	900	128	3	Simulated
Chinatown	20	345	24	2	Traffic
ChlorineConcentration	467	3840	166	3	Sensor
CinCECGTorso	40	1380	1639	4	Sensor
Coffee	28	28	286	2	Spectro
Computers	250	250	720	2	Device
CricketX	390	390	300	12	Motion
CricketY	390	390	300	12	Motion
CricketZ	390	390	300	12	Motion
Crop	7200	16800	46	24	Image
DiatomSizeReduction	16	306	345	4	Image
DistalPhalanxOutlineCorrect	600	276	80	2	Image
Earthquakes	322	139	512	2	Sensor
ECG200	100	100	96	2	ECG
ECG5000	500	4500	140	5	ECG
ECGFiveDays	23	861	136	2	ECG
ElectricDevices	8926	7711	96	7	Device
EOGVerticalSignal	362	362	1250	12	EOG
EthanolLevel	504	500	1751	4	Spectro
FaceAll	560	1690	131	14	Image

FaceFour	24	88	350	4	Image
FacesUCR	200	2050	131	14	Image
FiftyWords	450	455	270	50	Image
Fish	175	175	463	7	Image
FordA	3601	1320	500	2	Sensor
FreezerRegularTrain	150	2850	301	2	Sensor
GunPoint	50	150	150	2	Motion
Ham	109	105	431	2	Spectro
HandOutlines	1000	370	2709	2	Image
Haptics	155	308	1092	5	Motion
Herring	64	64	512	2	Image
HouseTwenty	34	101	3000	2	Device
InlineSkate	100	550	1882	7	Motion
InsectEPGRegularTrain	62	249	601	3	EPG
InsectWingbeatSound	220	1980	256	11	Sensor
ItalyPowerDemand	67	1029	24	2	Sensor
LargeKitchenAppliances	375	375	720	3	Device
Lightning2	60	61	637	2	Sensor
Lightning7	70	73	319	7	Sensor
Mallat	55	2345	1024	8	Simulated
Meat	60	60	448	3	Spectro
MedicalImages	381	760	99	10	Image
MelbournePedestrian	1200	2450	24	10	Traffic
MixedShapesRegularTrain	500	2425	1024	5	Image
MoteStrain	20	1252	84	2	Sensor
NonInvasiveFetalECGThorax1	1800	1965	750	42	ECG
NonInvasiveFetalECGThorax2	1800	1965	750	42	ECG
OSULeaf	200	242	427	6	Image
OliveOil	30	30	570	4	Spectro
PhalangesOutlinesCorrect	1800	858	80	2	Image
Plane	105	105	144	7	Sensor
PowerCons	180	180	144	2	Power
ProximalPhalanxOutlineCorrect	600	291	80	2	Image
RefrigerationDevices	375	375	720	3	Device
Rock	20	50	2844	4	Spectrum
ScreenType	375	375	720	3	Device
SemgHandGenderCh2	300	600	1500	2	Spectrum
ShapesAll	600	600	512	60	Image
SmoothSubspace	150	150	15	3	Simulated
SonyAIBORobotSurface1	20	601	70	2	Sensor
SonyAIBORobotSurface2	27	953	65	2	Sensor
StarLightCurves	1000	8236	1024	3	Sensor
Strawberry	613	370	235	2	Spectro
SwedishLeaf	500	625	128	15	Image
Symbols	25	995	398	6	Image
SyntheticControl	300	300	60	6	Simulated
ToeSegmentation1	40	228	277	2	Motion
Trace	100	100	275	4	Sensor
TwoLeadECG	23	1139	82	2	ECG
TwoPatterns	1000	4000	128	4	Simulated
UMD	36	144	150	3	Simulated
UWaveGestureLibraryX	896	3582	315	8	Motion
UWaveGestureLibraryY	896	3582	315	8	Motion

UWaveGestureLibraryZ	896	3582	315	8	Motion
Wafer	1000	6164	152	2	Sensor
Wine	57	54	234	2	Spectro
WordSynonyms	267	638	270	25	Image
Worms	181	77	900	5	Motion
Yoga	300	3000	426	2	Image

B.2 Proposed, non z-normalized, datasets

Table 4: New datasets collection: 35 datasets from both the UCR archive (dashed line) and the Monash UEA extrinsic regression archive. When missing values and/or varying lengths, replace missing values with 0 and pad series to maximum length with 0. All of the datasets are not z-normalized. AUC gain is mean improvement, aggregated over a time step window of length 5, e.g. in the first line, mean test AUC gets 11% better when using $[40\%, ..., 60\%]$ and 16% better with $[75\%, ..., 100\%]$ compared to $[5\%, ..., 25\%]$ of the series. *Italic* datasets are not included when classes are imbalanced as problems become too difficult for the chosen classifiers.

	Size	Length	Class	Type	AUC Gain	AUC Gain
					train	test
Data					(half/full)	(half/full)
BME	180	128	3	Simulated	(7%/7%)	(11%/16%)
Chinatown	365	24	2	Traffic	(1%/1%)	(0%/0%)
Crop	24000	46	24	Image	(9%/10%)	(8%/9%)
DodgerLoopDay	158	288	7	Sensor	(4%/5%)	(14%/17%)
EOGVerticalSignal	724	1250	12	EOG	(35%/35%)	(43%/45%)
GestureMidAirD1	338	360	26	Trajectory	(11%/12%)	(23%/26%)
GunPointAgeSpan	451	150	2	Motion	(7%/7%)	(7%/7%)
HouseTwenty	135	3000	2	Device	(1%/1%)	(5%/6%)
MelbournePedestrian	3650	24	10	Traffic	(15%/15%)	(15%/16%)
PLAID	1074	Vary	11	Device	(0%/0%)	(2%/2%)
Rock	70	2844	4	Spectrum	(1%/1%)	(12%/12%)
SemgHandGenderCh2	900	1500	2	Spectrum	(1%/2%)	(11%/13%)
SmoothSubspace	300	15	3	Simulated	(21%/37%)	(20%/38%)
UMD	180	150	3	Simulated	(10%/11%)	(22%/28%)
AcousticContaminationMadrid	138	365	2	Environment	(1%/2%)	(5%/7%)
AluminiumConcentration	629	2542	2	Environment	(3%/4%)	(5%/10%)
BitcoinSentiment	332	24	2	Sentiment	(11%/13%)	(-5%/-6%)
ChilledWaterPredictor	459	168	2	Energy	(0%/0%)	(8%/11%)
CopperConcentration	629	2542	2	Environment	(4%/5%)	(4%/3%)
Covid19Andalusia	204	91	2	Health	(7%/8%)	(7%/17%)
DailyOilGasPrices	188	30	2	Economy	(25%/23%)	(10%/8%)
DhakaHourlyAirQuality	2068	24	2	Environment	(2%/3%)	(0%/2%)
ElectricityPredictor	810	168	2	Energy	(4%/5%)	(7%/13%)
FloodModeling3	613	266	2	Environment	(20%/22%)	(20%/31%)
HouseholdPowerConsumption1	1431	1440	2	Energy	(4%/7%)	(14%/27%)
HotwaterPredictor	351	168	2	Energy	(1%/1%)	(4%/5%)
MadridPM10Quality	6923	168	2	Environment	(4%/5%)	(11%/16%)
ParkingBirmingham	1888	14	2	Environment	(7%/28%)	(5%/22%)
PrecipitationAndalusia	672	365	2	Environment	(0%/1%)	(3%/4%)
SierraNevadaMountainsSnow	500	30	2	Environment	(6%/7%)	(-2%/4%)
SolarRadiationAndalusia	672	365	2	Energy	(1%/1%)	(4%/4%)
SteamPredictor	300	168	2	Energy	(2%/2%)	(3%/6%)
TetuanEnergyConsumption	364	144	2	Energy	(5%/5%)	(1%/6%)
WindTurbinePower	852	144	2	Energy	(11%/12%)	(15%/21%)

C Supplementary results

C.1 Additional figures : Standard cost setting

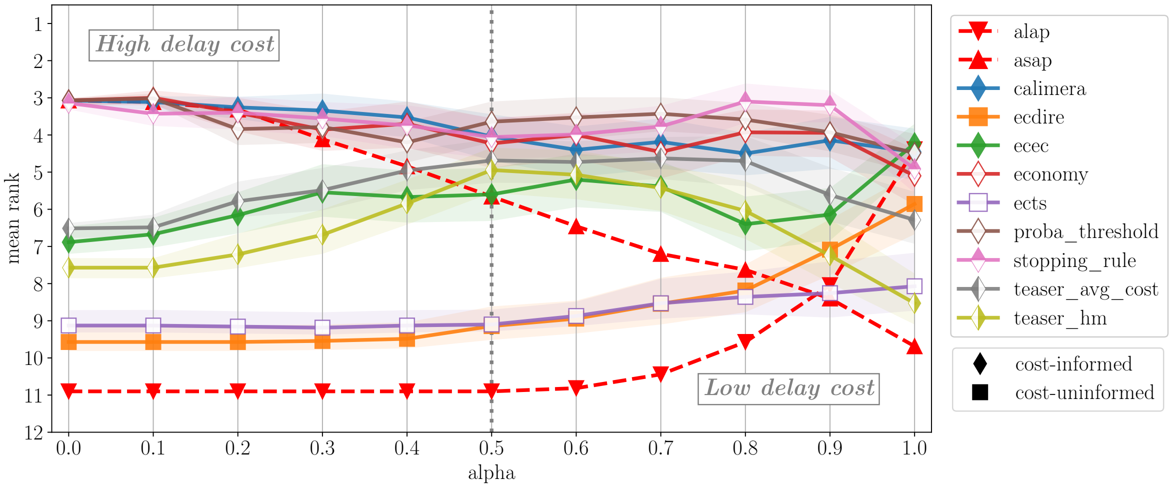


Figure 9: Standard cost setting, non z-normalized proposed datasets (orange cylinder). Compared to Figure 4a, the global ranking is not altered much. One can observe that for $\alpha \in [0.5, 0.7]$ the top group is now more populated, gathering the first six approaches, probably due to the limited amount of datasets available in this case.

C.2 Additional figures : Anomaly detection cost setting

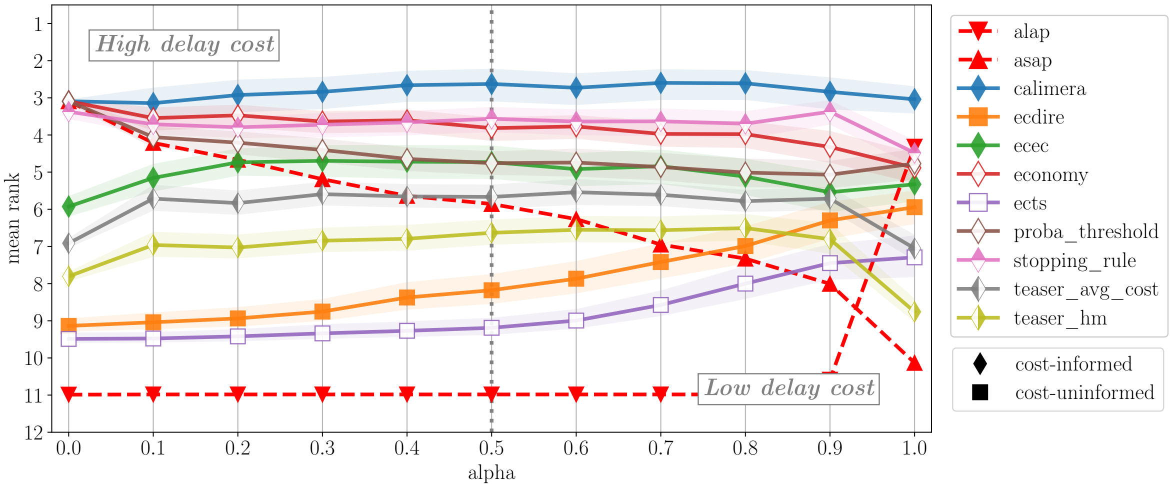


Figure 10: Anomaly detection cost setting, original UCR datasets (blue cylinder). Compared to Figure 7a, one can see that Calimera is now clearly dominating all other methods for all α. The global ranking remains globally stable otherwise.

C.3 Removing calibration

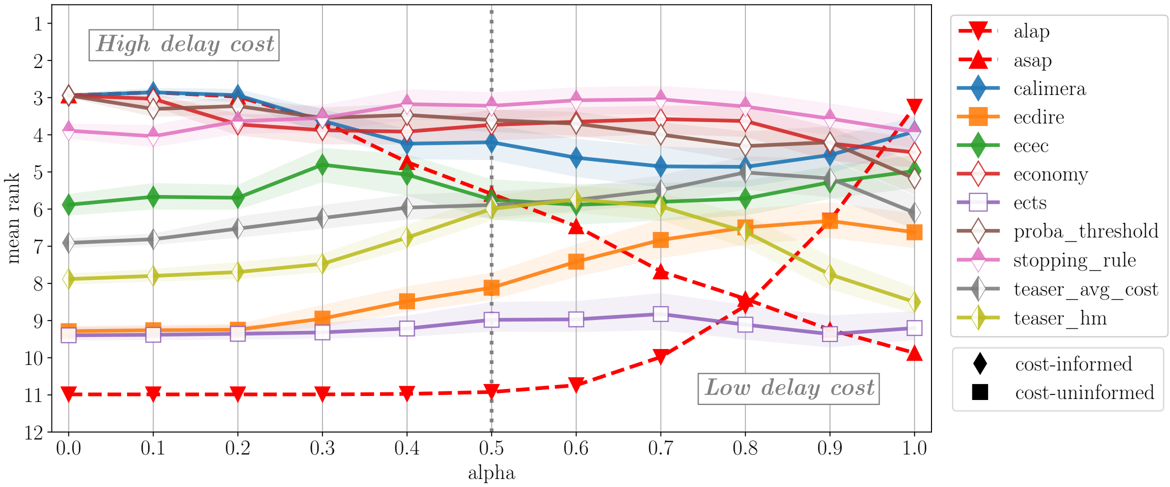


Figure 11: Standard cost setting, original UCR datasets (blue cylinder). The calibration step is now removed, i.e. the outputs from the decision function is now simply passed through a *softmax* function. Both Calimera and Proba Threshold suffer heavily from using uncalibrated scores.

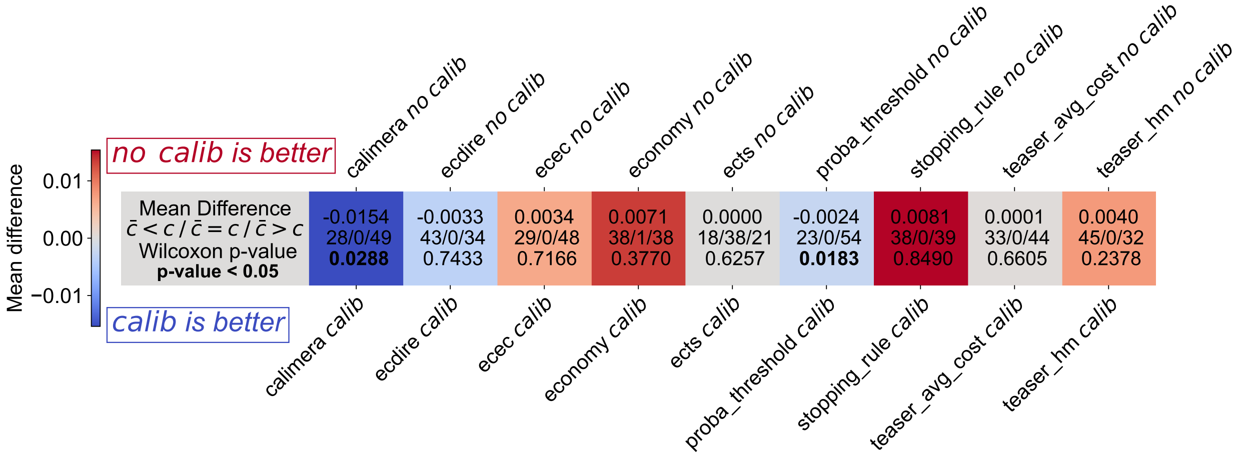


Figure 12: Pairwise comparison (Ismail-Fawaz et al., 2023), calibration (*calib* / C) vs no calibration (*no calib* / $\bar{C}$). We select $\alpha = 0.8$ as the alpha value where both the naive baselines cross, i.e. where, in average, most of datasets are more challenging. Square colors are indexed on the mean *AvgCost* difference. For example, Calimera has a lower mean *AvgCost* when trained over calibrated scores: it appears in dark blue. The Wilcoxon p-value is equal to 0.0288, which is lower than significance level equal to 0.05. Thus, Calimera statistically under-performs when using uncalibrated scores. This is also the case for the Proba Threshold method.

C.4 Changing the base classifier

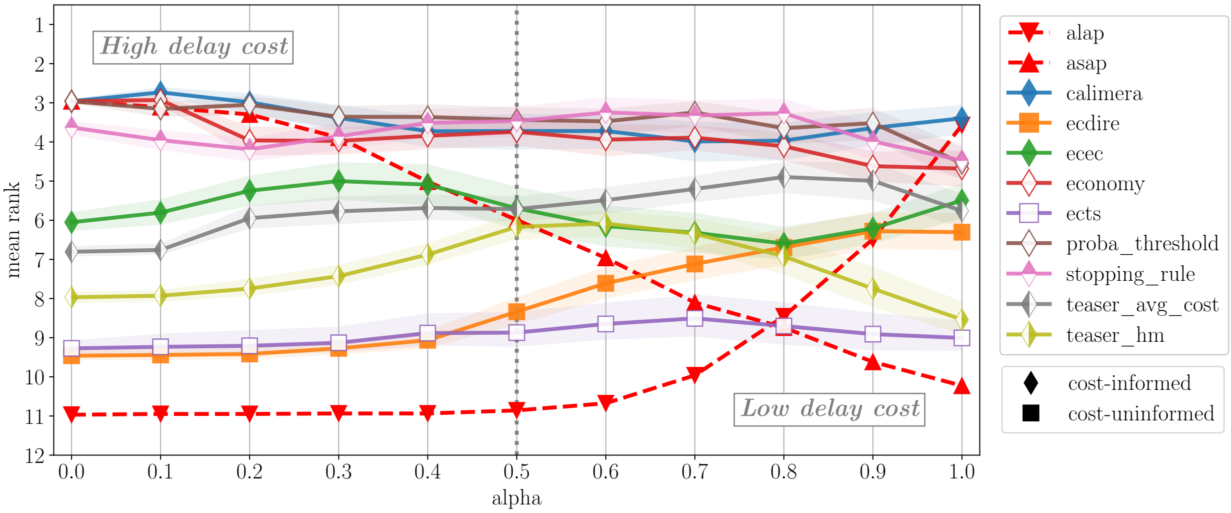


Figure 13: Standard cost setting, original UCR datasets (blue cylinder). The base classifier is now WEASEL 2.0 Schäfer & Leser (2023). Results are very close to those exposed in Figure 4a.

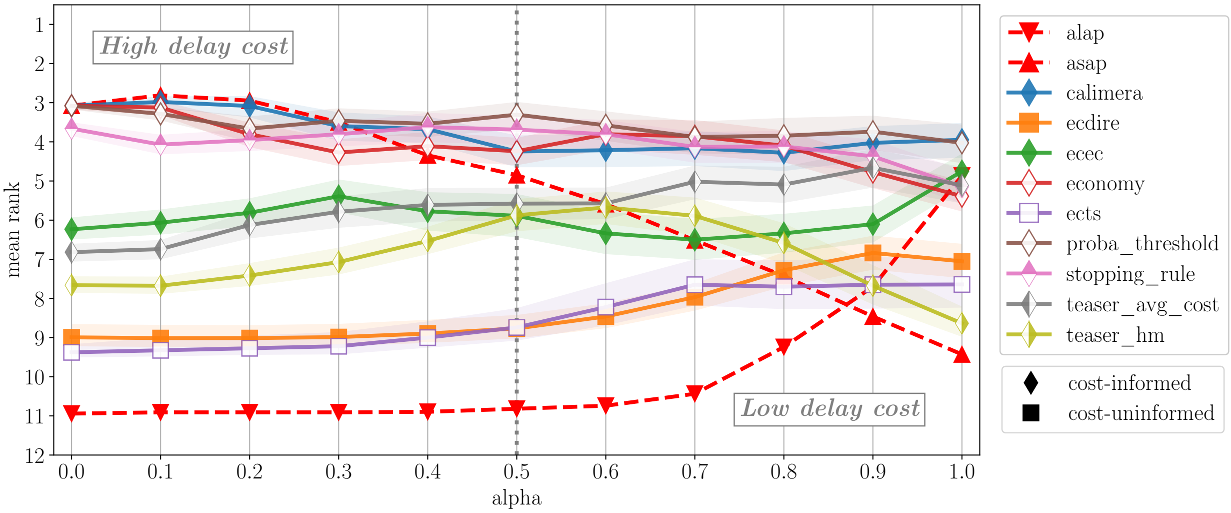


Figure 14: Standard cost setting, original UCR datasets (blue cylinder). The base classifier is now a pipeline including features extraction with TSFRESH (Christ et al., 2018) and classification using XGBOOST (Chen et al., 2015). Results are a bit noisier than those exposed in Figure 4a.

Table 5: Comparison of the tested classifiers. Percentage representing, for each alpha, the amount of dataset for which each classifier is ranked first, averaged over all trigger models and all datasets. Ties are not considered ; thus, each line may sum to less than 1. Best performing classifier is underlined.

	classifier		
α	MiniROCKET	Weasel 2.0	tsfresh&XGBoost
0	12.60%	11.95%	16.36%
0.1	31.56%	18.44%	37.01%
0.2	32.86%	18.96%	36.49%
0.3	36.49%	19.48%	32.34%
0.4	39.87%	17.53%	31.30%
0.5	43.12%	19.10%	26.62%
0.6	42.60%	23.25%	23.51%
0.7	44.16%	24.94%	20.52%
0.8	48.44%	25.45%	15.71%
0.9	48.70%	26.62%	14.29%
1	42.21%	26.75%	14.68%

C.5 Impact of z-normalization

Clearly, using z-normalized datasets is not applicable in practice, as it would require knowledge of the entire incoming time series. In a research context, previous work has used such training sets to test the proposed algorithms. Our goal here, is to assess whether this could have a large impact on the performances. For example, when a normalized time series has a low variance at the beginning, we can expect a high variance in the rest of the series since the mean variance is 1. There is therefore an information leakage that can be exploited by an ECTS algorithm, while this is not representative of what happens in real applications. A proposal such as the one presented by Schäfer & Leser (2020), where the z-normalization of available time series is repeated at each time step, has its own problems. In particular, it means that if a single classifier is used for all time steps, the representation of $\mathbf{x}_t$ can be different at times t and $t+1$ and all further time steps which can induce confusion for the classifier.

On the one hand, z-normalization induces an information leakage that could help methods to unduly exploit knowledge about the future of incoming time series. On the other hand, any normalization rescales the signal and therefore, potentially, hinder the recognition of telltale features. So, does z-normalization affect the performance of ECTS methods? And if yes, in which way?

In order to answer this question, we took the new datasets collection described in Section 4.1. They are indeed not z-normalized originally. We duplicate and z-normalized them to get a second collection. As explained in Section 4.1, some extrinsic regression datasets have been converted into classification ones. Here, the threshold value chosen to discretize the output into binary classes has been set to the median of the regression target. In this way, classes within those datasets are equally populated.

In these experiments, the delay cost is linear as in Section 4.2 and as in most of the literature. Figure 15 reports pairwise comparisons done on the 35 datasets. We look at $\alpha = 0.8$, as this is the only value for which significant differences are observed. One can see that most of the trigger models do not actually benefit from the z-normalization. Quite the opposite: out of nine trigger models, only one, i.e. Ecdire, actually has a better mean *AvgCost* when being trained on z-normalized data. Regarding the remaining methods, both Stopping Rule and Teaser$_{Avg}$ perform significantly worse when operating on z-normalized data. Those trends are quite similar for other α values, without any significance on the statistical tests though. Thus, while z-normalization has some impact, since privileged information from the future can be leaked, our experiments, for the proposed datasets collection at least, show that this does not alter the overall results reported in the literature, and are globally in accordance with the results presented in Section 4.

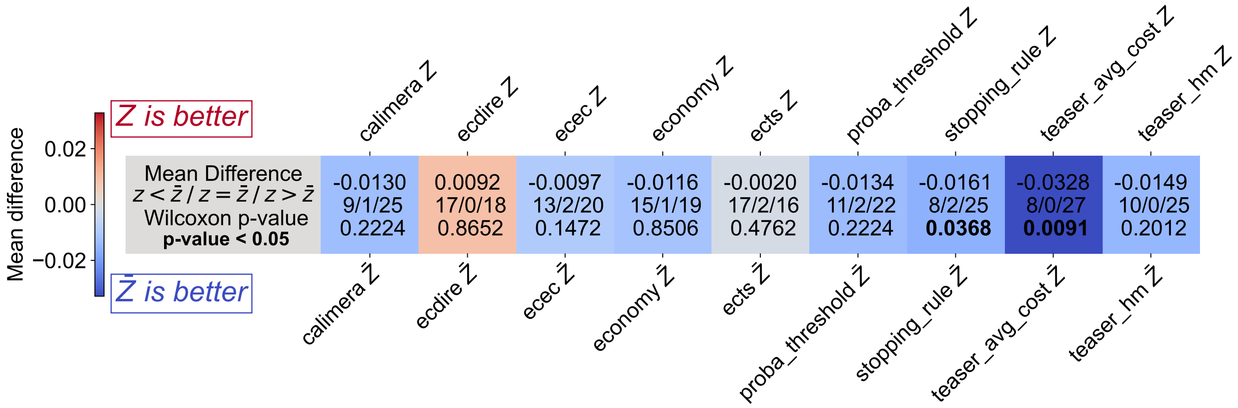
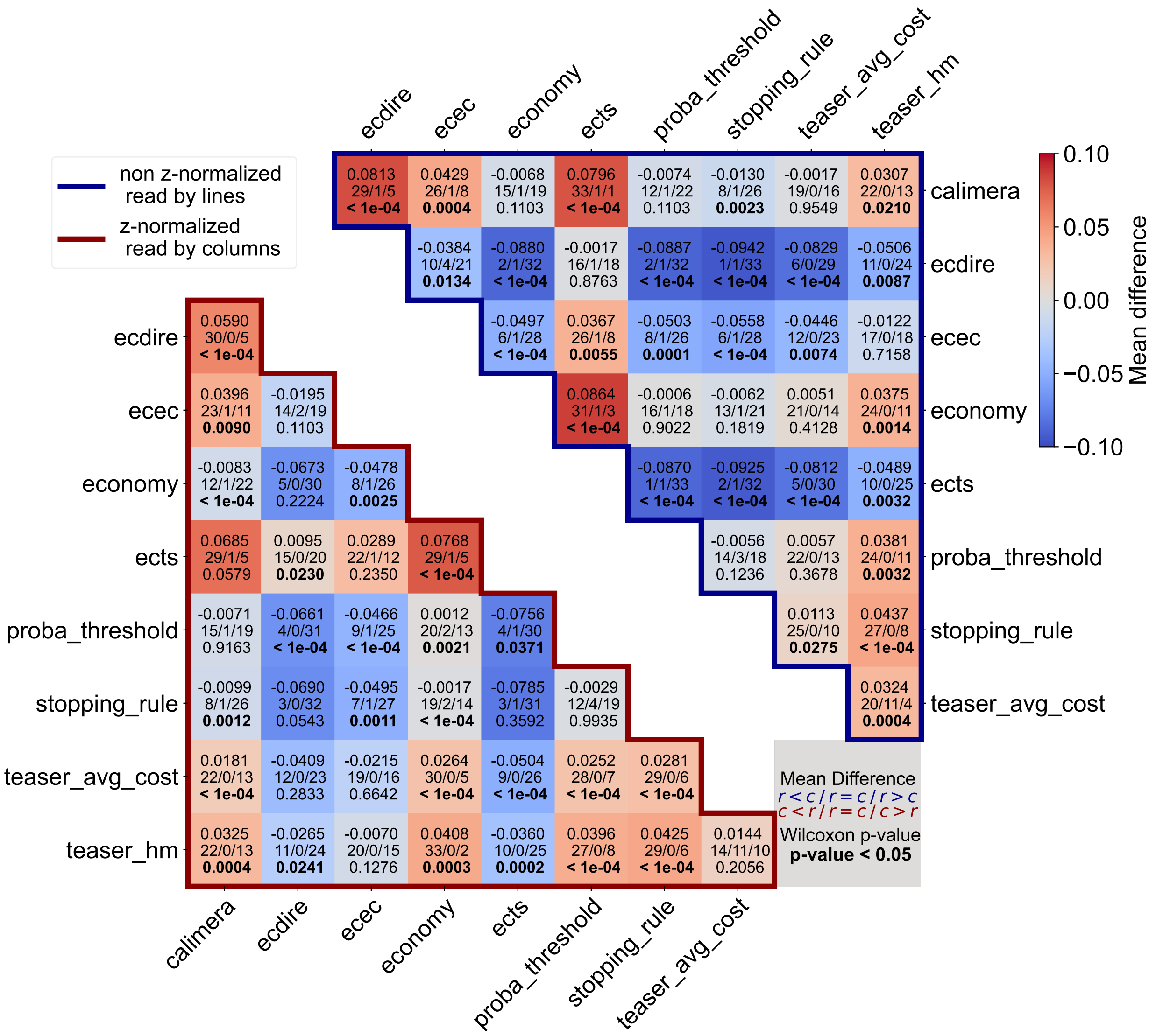


Figure 15: Pairwise comparison (Ismail-Fawaz et al., 2023), z-normalization (Z) vs no z-normalization ($\bar{Z}$), $\alpha = 0.8$. Square colors are indexed on the mean *AvgCost* difference. For example, Calimera has a lower mean *AvgCost* when trained over non z-normalized datasets by 1.3e-2 and appears in light blue. It beats the z-normalized version over 25 datasets, loses over 9 and are tied on 1. The Wilcoxon p-value is equal to 0.2224, which is higher than significance level equal to 0.05. Thus, no statistical difference can be observed for the considered approach.



	ecdire	ecec	economy	ects	proba_threshold	stopping_rule	teaser_avg_cost	teaser_hm
calimera	0.0813 29/1/5 **< 1e-04**	0.0429 26/1/8 **0.0004**	-0.0068 15/1/19 0.1103	0.0796 33/1/1 **< 1e-04**	-0.0074 12/1/22 0.1103	-0.0130 8/1/26 **0.0023**	-0.0017 19/0/16 0.9549	0.0307 22/0/13 **0.0210**
ecdire		-0.0384 10/4/21 **0.0134**	-0.0880 2/1/32 **< 1e-04**	-0.0017 16/1/18 0.8763	-0.0887 2/1/32 **< 1e-04**	-0.0942 1/1/33 **< 1e-04**	-0.0829 6/0/29 **< 1e-04**	-0.0506 11/0/24 **0.0087**
ecec			-0.0497 6/1/28 **< 1e-04**	0.0367 26/1/8 **0.0055**	-0.0503 8/1/26 **0.0001**	-0.0558 6/1/28 **< 1e-04**	-0.0446 12/0/23 **0.0074**	-0.0122 17/0/18 0.7158
economy				0.0864 31/1/3 **< 1e-04**	-0.0006 16/1/18 0.9022	-0.0062 13/1/21 0.1819	0.0051 21/0/14 0.4128	0.0375 24/0/11 **0.0014**
ects					-0.0870 1/1/33 **< 1e-04**	-0.0925 2/1/32 **< 1e-04**	-0.0812 5/0/30 **< 1e-04**	-0.0489 10/0/25 **0.0032**
proba_threshold						-0.0056 14/3/18 0.1236	0.0057 22/0/13 0.3678	0.0381 24/0/11 **0.0032**
stopping_rule							0.0113 25/0/10 **0.0275**	0.0437 27/0/8 **< 1e-04**
teaser_avg_cost								0.0324 20/11/4 **0.0004**

	calimera	ecdire	ecec	economy	ects	proba_threshold	stopping_rule	teaser_avg_cost
ecdire	0.0590 30/0/5 **< 1e-04**							
ecec	0.0396 23/1/11 **0.0090**	-0.0195 14/2/19 0.1103						
economy	-0.0083 12/1/22 **< 1e-04**	-0.0673 5/0/30 0.2224	-0.0478 8/1/26 **0.0025**					
ects	0.0685 29/1/5 0.0579	0.0095 15/0/20 **0.0230**	0.0289 22/1/12 0.2350	0.0768 29/1/5 **< 1e-04**				
proba_threshold	-0.0071 15/1/19 0.9163	-0.0661 4/0/31 **< 1e-04**	-0.0466 9/1/25 **< 1e-04**	0.0012 20/2/13 **0.0021**	-0.0756 4/1/30 **0.0371**			
stopping_rule	-0.0099 8/1/26 **0.0012**	-0.0690 3/0/32 0.0543	-0.0495 7/1/27 **0.0011**	-0.0017 19/2/14 **< 1e-04**	-0.0785 3/1/31 0.3592	-0.0029 12/4/19 0.9935		
teaser_avg_cost	0.0181 22/0/13 **< 1e-04**	-0.0409 12/0/23 0.2833	-0.0215 19/0/16 0.6642	0.0264 30/0/5 **< 1e-04**	-0.0504 9/0/26 **< 1e-04**	0.0252 28/0/7 **< 1e-04**	0.0281 29/0/6 **< 1e-04**	
teaser_hm	0.0325 22/0/13 **0.0004**	-0.0265 11/0/24 **0.0241**	-0.0070 20/0/15 0.1276	0.0408 33/0/2 **0.0003**	-0.0360 10/0/25 **0.0002**	0.0396 27/0/8 **< 1e-04**	0.0425 29/0/6 **< 1e-04**	0.0144 14/11/10 0.2056

Figure 16: Multi-comparison-matrices (Ismail-Fawaz et al., 2023). The upper triangle, with dark blue contours, displays the comparison of the competitive methods trained over non z-normalized dataset (orange cylinder). The values within this triangle has to be read *by lines*, i.e. for a considered line, red shades indicate better performances, blue shades weaker performances. The lower triangle, with dark red contours, is the comparison of the methods trained over the same datasets, z-normalized (chocolate cylinder). The values within this triangle has to be read *by columns*, i.e. for a considered column, red shades indicate better performances, blue shades weaker performances. The complete figure being symmetrical indicates that z-normalization does not impact much relative ranking between methods.

C.6 Anomaly detection cost setting : an ablation study

Exponential delay cost only

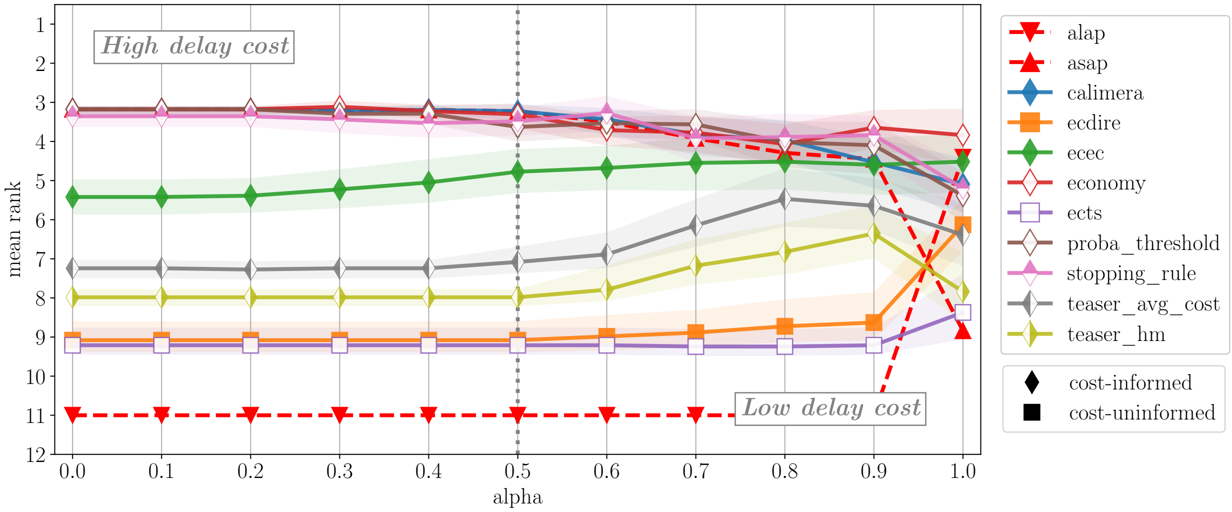


Figure 17: Exponential delay cost, symmetric binary misclassification cost, non z-normalized proposed imbalanced datasets (orange cylinder with a whole).

Imbalanced misclassification cost only

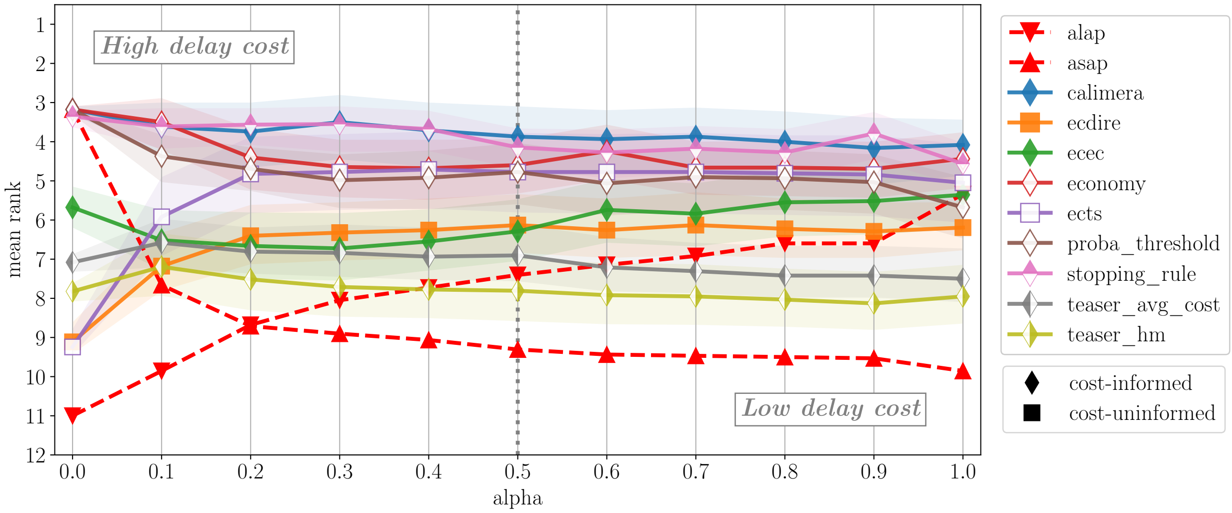


Figure 18: Linear delay cost, non symmetric imbalanced misclassification cost, non z-normalized proposed imbalanced datasets (orange cylinder with a whole).

Standard cost setting, imbalanced datasets

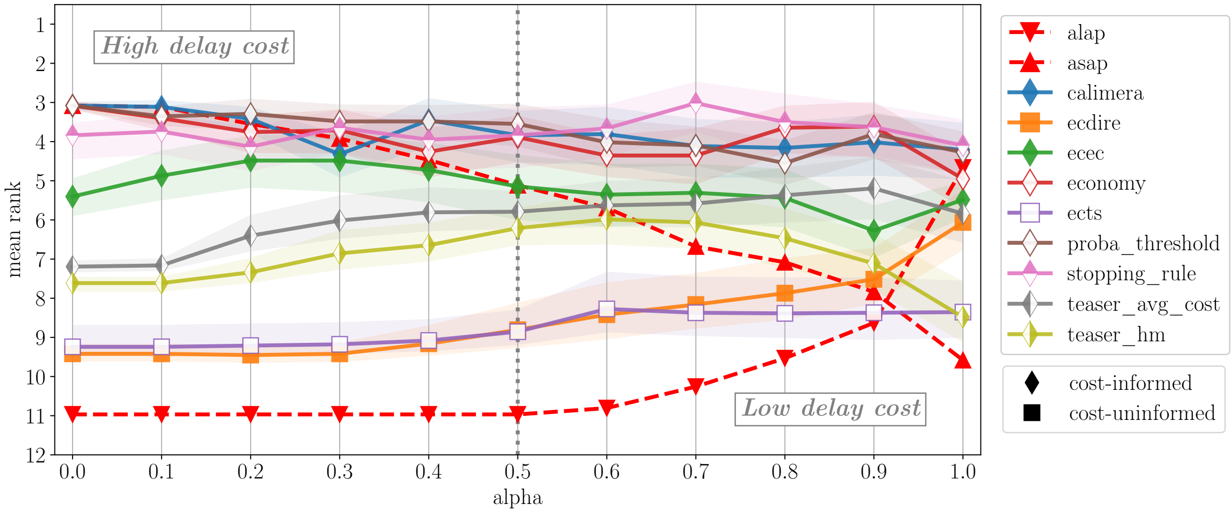


Figure 19: Linear delay cost, symmetric binary misclassification cost, non z-normalized proposed imbalanced datasets (orange cylinder with a whole).