Received 10 October 2022, accepted 4 November 2022, date of publication 24 November 2022, date of current version 9 December 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3224467

RESEARCH ARTICLE

Measuring Early Detection of Anomalies

MANUEL F. LÓPEZ-VIZCAÍNO, FRANCISCO J. NOVOA, DIEGO FERNÁNDEZ, AND FIDEL CACHEDA

Center for Information and Communications Technologies Research (CITIC), Department of Computer Science and Information Technologies, University of A Coruña, 15071 A Coruña, Spain

Corresponding author: Manuel F. López-Vizcaíno (manuel.fernandezl@udc.es)

This work was supported in part by the Ministry of Economy and Competitiveness of Spain and Fondo Europeo de Desarrollo Regional (FEDER) Funds of the European Union under Project PID2019-111388GB-I00; and in part by the Centro de Investigación de Galicia—Centro de Investigación en Tecnologías de la Información y las Comunicaciones (CITIC) Funded by Xunta de Galicia and the European Union (European Regional Development Fund–Galicia 2014-2020 Program), under Grant ED431G 2019/01.

ABSTRACT Early detection is a matter of growing importance in multiple domains as network security, health conditions over social network services or weather forecasts related disasters. It is not enough to make a good decision but it also needs to be made on time. In this paper, we define a method to evaluate detection of anomalies in time-aware systems. To do so, we present the early detection problem from a generic perspective, examine the evaluation metrics available and propose a new metric, named *TaP* (*Time aware Precision*). A set of experiments using three different datasets from different fields are performed in order to compare the behaviour of the different metrics. Two different approaches were followed, first a batch evaluation is performed, followed by a streaming evaluation which allows to present a more realistic behaviour of the systems. For both steps, we propose two sets of experiments. The first one using baseline models, followed by the evaluation of a set of Machine Learning algorithms results. The presented metric allows the amount of items needed to take a decision to be taken into account, not depending on the specific dataset but on the nature of the problem to solve.

INDEX TERMS Early detection, machine learning, time-aware metrics, real-time systems, classification algorithms, network security, social network services.

I. INTRODUCTION

The detection of anomalies in multiple application domains, such as health conditions, cyber-security or industrial equipment malfunction, is a key issue that has attracted, and continues to attract a lot of attention, using multiple and different approaches, such as, statistical, classification or cluster based, among others [1].

However, another key aspect behind the detection of an anomaly is the time required for its detection, since an early detection can help reduce the negative impact of the anomaly in the system. For example, from a medical perspective, the early detection of a disease can speed up its treatment reducing the negative impact of the disease on the subject and, at the same time, reducing the economical cost of the treatment [2].

The associate editor coordinating the review of this manuscript and approving it for publication was Mehul S. Raval.

In the same way, the early detection of an anomaly in an industrial equipment can help minimise the disruption of a normal service operation [3], [4]. The early detection of an intrusion in a computer network is paramount to reduce the impact of the attack on the infrastructure and to prevent it from reaching more advanced phases, where it could be more difficult to tackle [5].

Therefore, in this work, we study the problem of an early detection of anomalous behaviour in different environments, following an evaluation methodology that takes into consideration the time required to detect the anomaly. Then, we aim at classifying as soon as possible, minimising the amount of information considered by the models to make a decision. For this purpose, we examine the main works in the state-of-the-art that tackle the early detection problem in different environments and we propose a formal and common structure for the early detection problem, along with a suitable evaluation metric.

More specifically, the main contributions of this work can be summarised as follows:

- We present and formally define the early detection problem from a global perspective, which seeks to identify as soon as possible, if an anomaly is present in the system.
- We examine the evaluation metrics from the state-of-the-art, identify pros and cons, and we propose a new metric named *Time aware Precision*, that covers the three states in the early detection problem (i.e. normal, anomalous, delay).
- We perform extensive experiments using three independent real-world datasets from different environments (depression disorders, computer network attacks and floods) and we validate the behaviour of the metric proposed over the state-of-the-art metrics.

The remainder of this article is organised as follows.

First, the related studies are examined regarding the early detection problem in different contexts. Then, we formally define the early detection problem from a global perspective and next we present the datasets used in our experiments that cover different environments of the problem. A detailed experimental evaluation is described in Section V, followed by a discussion. Finally, we present the main conclusions of our article and possible future works.

II. RELATED WORKS

Throughout the literature, early detection of anomalies has been explored in different fields, although there is limited research on early detection evaluation metrics and methodologies.

The workshop on early risk prediction on the Internet (eRisk), as part of the Conference and Labs for the Evaluation Forum (CLEF) since 2017, proposed a task for the early detection of different conditions (e.g. depression, self-harm or anorexia) on social networks using a time-aware methodology and effectiveness metrics [6], [7].

In this sense, to the best of our knowledge, the eRisk workshop constitutes the best effort on early detection with different metrics proposed, being *ERDE* (Early Risk Detection Error) [6] and latency-weighted F1 (or $F1 - latency$) [8] the most used. These metrics are described in detail in the next section, although they both present a major issue that is to be dataset dependent, which prevents their use for performance comparison. In fact, this limitation on these metrics is the main motivation behind our work.

On the other side, several attempts for early detection in different fields can be identified in the literature. For example, some works explore the early detection of cyberbullying on social networks. Samghabadi et al. [9] propose a corpus to detect cyberbullying on social networks as soon as possible, although the time-aware evaluation is limited to the use of standard metrics (i.e. F1) at specific points. A more precise study is described in [10], where the authors use *ERDE* and latency-weighted $F1$ to compare the performance of different machine learning methods in a dataset from the Vine social network.

Some efforts have also been given to the early detection of rumours and fake news on social media. For example, [11] presents different alternatives for the detection of unconfirmed information, although the time-aware evaluation is limited to measuring the amount of time required to detect a rumour. An interesting interdisciplinary study of fake news is presented in [12], although the analysis of the early detection is circumscribed to the reduction of news articles and news content information, without including a proper time-aware evaluation.

Cyber-security is another field where the time required to detect a possible threat has been of interest for some researchers. In [13] the authors present a prototype for cyber attacks early recognition system, although no time-aware performance evaluation of the proposed system is developed. Some other works, such as [14] or [15], explore early detection by detecting cyber attacks in the early stages, however the evaluations do not actually consider the time required for the detection. There is some research on Early Warning Systems (EWS), especially to avoid malware propagation, that explore different alternatives such as bayesian inference [16], Kalman filter [17] or sensors [18], but the evaluation is mainly focused on the identification of potential attacks in a timeline, without presenting a proper time-aware performance metric. More closely related to this work, [19] explores different methods for the early detection of cyber attacks using *ERDE* as the main performance metrics, while on [20] the authors focus on Operating System scan attacks and include $F1 - latency$ as time-aware evaluation metric.

Another area in which early detection is especially interesting corresponds to smart cities. The number of connected devices and the massive traffic provoke that threats can have a significant impact. Xu in his work [21] proposes the use of software-defined network function virtualization (SDNFV) architecture and a traffic classification strategy to perform early detection of such attacks. However, the time-aware evaluation is limited to measuring the average response time between the attack and the detection. A similar approach is followed by Privalov et al. [22], measuring the average time to detect a distributed denial of service attack.

In summary, in the literature we can identify several research works that emphasise the early detection of problems or anomalies in different fields. Particularly we should highlight *ERDE* [6] and $F_{latency}$ [8]. However, there is not a clear time-aware evaluation methodology, which leads to the use of unconventional alternatives and metrics by the researchers. In this work, we present, from a global perspective, the early detection of anomalies problem and propose a new metric, named *Time aware Precision*, generic and suitable for any situation.

III. PROBLEM STATEMENT

In this section we formally define the problem of early detection of anomalous situations for a generic system and a non-specific type of entity.

Let $E = \{e_1, e_2, \ldots, e_{|E|}\}$ be the set of entities that may suffer anomalies, where $|E|$ denotes the number of entities. Each entity $e \in E$ is formed of a sequence of items, denoted as I_e, and a binary indicator l_e that denotes whether the specific entity is considered anomalous ($l_e = true$) or not ($l_e = false$).

We define E^+ as the set of anomalous entities, i.e. where $l_e = true$, and E^- as the set of non-anomalous entities:

$$E^+ = \{e_1^+, e_2^+, \ldots, e_{|E^+|}^+\}, \quad \forall\, e_i^+ \in E^+\ l_{e_i^+} = true$$
$$E^- = \{e_1^-, e_2^-, \ldots, e_{|E^-|}^-\}, \quad \forall\, e_i^- \in E^-\ l_{e_i^-} = false$$

The sequence of items for a specific entity is supposed to change through time and is given by $I_e = (< I_1^e, t_1^e >, < I_2^e, t_2^e >, \ldots, < I_n^e, t_n^e >)$, where the tuple $< I_k^e, t_k^e >$, $k \in [1, n]$ represents the k-th item for entity e, and t_k^e is the timestamp associated with item I_k^e.

Timestamps $t_1^e, t_2^e, \ldots, t_n^e$ can be equally and homogeneously distributed (e.g. in case of sensor emitting temperature and humidity data every second) or can be uneven and randomly distributed (e.g. in case of the posts written by a user).

An item, I_k^e, is specified by a vector of characteristics or features and, in this case, we assume that all items associated with an entity, $I_k^e, k \in [1, n]$, are defined by the same vector of features, whose values may, and predictably will, change through time.

$$I_k^e = \left[f_{k_1}^e, f_{k_2}^e, \ldots, f_{k_m}^e\right], k \in [1, n]$$

Since entities are independent, each sequence of items I_e may have different lengths, n, for each entity $e \in E$. However, note that the number of features, m, would be the same for all items.

We will drop the superscript e in the specification of I_e, e.g. $I_e = (< I_1, t_1 >, < I_2, t_2 >, \ldots, < I_n, t_n >)$, whenever e is clear from the context. Similarly, the vector of features for a specific item I_k^e, will drop the superscript when it is clear from the context: $I_k^e = \left[f_{k_1}, f_{k_2}, \ldots, f_{k_m}\right]$.

Given an entity e, the objective is to detect if the entity has an anomalous behaviour but scanning as few items from I_e as possible.

We define a function $f(l_e, I_e \times [1..n]) \rightarrow \{0, 1, 2\}$ as the objective function. The function will return 1 (i.e. *positive*) if entity e is considered anomalous after processing items I_1 to I_k. In case the entity e is considered normal (i.e. non-anomalous or *negative*) after processing item k and previous ones, then $f(l_e, I_e, k) = 0$. Finally, $f(l_e, I_e, k) = 2$ if no definitive decision can be emitted on entity e after reading k items and more items must be processed (i.e. *delay*).

Therefore, for $f(l_e, I_e, k)$ outputs 0 and 1 are considered final and items $I_{k+1}, \ldots, I_n$ do not need to be processed. On the other side, if output 2 is provided, further items $I_{k+1}, \ldots, I_n$ must be processed, until a final output is achieved or the end of the items sequence is reached.

A. STANDARD METRICS

The main metrics identified in the state-of-the-art for early detection problems are *ERDE* [6] and latency-weighted F1 [8]. In both cases, metrics were used to measure performance on the early detection of depression on individuals based on their posts on social networks.

The *ERDE* metric is measured at a specific point denoted as o and considers four different cases [6]:

$$ERDE_o(e_i, k) = \begin{cases} \dfrac{\sum_{e_i \in E \wedge l_{e_i} = true} 1}{|E|} & \textit{if FP} \\ 1 & \textit{if FN} \\ 1 - \frac{1}{1+e^{k-o}} & \textit{if TP} \\ 0 & \textit{if TN} \end{cases}$$

In case of wrong predictions (false positive, FP, and false negative, FN), as expected, the error increases but in two different ways. False positives increase the error proportionally to the number of positive cases in the dataset, while false negatives increase the error by 1. A true negative (TN) prediction, as expected, does not increase the error independently of when it was produced. However, a true positive (TP) will impact negatively if the delay required to make the prediction exceeds the measuring point o (i.e. $k > o$), using a sigmoid function to introduce the penalty.

Note that *ERDE* metric ranges from 0 to +1 and, as an error measure, values closer to zero are considered preferable values.

Also, some variants of the *ERDE* metric can be found in the literature. For example, [23] defines the $ERDE_o^{\%}$ that is based on the percentage of items processed, instead of the number of items and in [20], a normalized version of the *ERDE* metric is defined using mix-max normalization.

The latency-weighted F1, or in short $F_{latency}$ or *F1-latency*, is proposed by Sadeque et al. as an alternative to the *ERDE* metric, combining both latency and accuracy [8]. The F-latency metric is defined as:

$$F_{latency}(e_i, k) = F1 \cdot \left(1 - \underset{l_{e_i} = true}{median}\left(-1 + \frac{2}{1 + e^{-p\cdot(k-1)}}\right)\right)$$

where, $F1$ is the standard *F-measure* that is calculated based on precision (P) and recall (R), $F1 = \frac{2 \cdot P \cdot R}{P \cdot R}$, and p is a parameter that determines how quickly the penalty should increase. Typically, this parameter is set to achieve 50% of latency penalty at the median number of items.

This metric, as it depends directly from $F1$, will range from 0 to 1 and values closer to 1 are representative of good results.

The main limitation that both metrics present is that they are dataset dependent, which greatly limits the performance comparison among different datasets. The *ERDE* metric penalisation for false positives is directly related with the number of anomalous cases in the dataset and for the $F_{latency}$ the parameter p depends on the items median.

Moreover, some other limitations have been identified for both metrics:

- The sigmoid function employed by *ERDE* produces a fast increase of the penalty for late true positives. Also, a perfect system (detecting all true positive cases just with the first round of items), may achieve an error greater than 0 [20], [24].
- $F_{latency}$ is defined as a final metric and cannot be measured at different points (i.e. different batches) and it can produce unexpected values (i.e. negative values) with small item sequences due to the use of the median to determine the penalty increase in the parameter p [20].

B. TIME AWARE PRECISION

One of the main contributions of this work is a new metric named *Time aware Precision* or *TaP* in short. This metric is expected to overcome all major issues of *ERDE* and $F_{latency}$.

TaP will range between +1 and −1, where values close to +1 represent correct and in time predictions, values close to −1 represent incorrect or late predictions and values around 0 would correspond to non predictions (or delay) cases.

We define *Time aware Precision* as a generic metric that could be applied to any early detection problem. In this sense, one of the key aspects that is problem dependent, is the moment when a correct prediction is considered late. For this purpose, and following *ERDE*, we define a point denoted as o to start penalising correct predictions. Up to this point it will be considered as an acceptable delay for the defined problem and the penalisation will be applied after that amount of items.

Therefore, *TaP* at o for an entity e_i is calculated as follows:

$$TaP_{o,\lambda}(e_i,k) = \begin{cases} -1 & \textit{if } FP \vee FN \\ 1 & \textit{if } TP \vee TN \wedge k \leq o \\ 1 - pf_{o,\lambda}(k) & \textit{if } TP \vee TN \wedge k > o \\ 0 & \textit{if delay} \end{cases}$$

In the case of an incorrect prediction (False Positive or False Negative), the metric weighs −1 to represent an error. If a correct prediction is made (True Positive or True Negative), then the time required to generate the prediction is taken into account. If the prediction was made before o (i.e. $k \leq o$), then the precision achieves its maximum value (i.e. 1). Otherwise (i.e. $k > o$) a penalty function, $pf_{o,\lambda}(k)$, is used to reduce the score of the metric. Lastly, if no decision has been made (i.e. *delay*), *TaP* will take the value of 0.

TaP for the set of entities E is calculated as the average score for each entity:

$$TaP_{o,\lambda}(E,k) = \frac{1}{|E|}\sum_{e_i \in E} TaP_{o,\lambda}(e_i,k)$$

The penalty function is defined using a generalised logistic function scaled to operate on the defined range and the penalty is based on the number of items required to take the decision:

$$pf(k)_{o,\lambda} = 2 \cdot \left(-1 + \frac{2}{1+e^{-\lambda(k-o)}}\right)$$

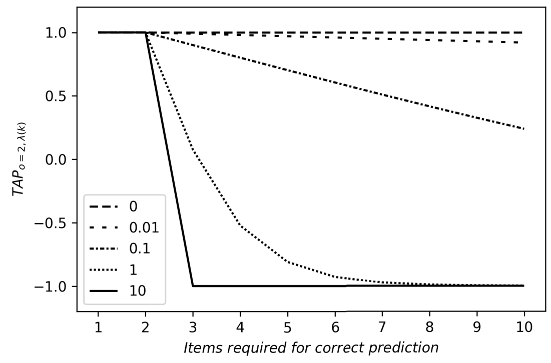


FIGURE 1. ***TaP* for $o = 2$ and different values of λ. The X-axis represents the number of items required by the system to reach a correct prediction, assuming all previous items led to a delay. For example, for $x = 5$ the system generated the correct prediction on item $k = 5$ and, therefore, previously (i.e. $x < 5$) a delay was generated.**

Where the parameter λ controls how the penalty is increased.

Figure 1 shows how parameter λ affects *TaP*. In all cases, the correct prediction has been made, but the number of items required to obtain it varies and is represented on the X-axis. If $\lambda = 0$ then no penalty is introduced for a late prediction and, independently of when the prediction was made, the maximum score is achieved. However, as λ increases the penalty slope becomes steeper. When $\lambda = 0.1$ the metric decreases linearly as the correct prediction is being delayed. If $\lambda = 10$ a step penalty function is obtained, reaching the minimum value with a prediction just one item late (i.e. $k = 3$).

Also, we define, respectively, TaP^+ for the set of anomalous entities or positives cases, E^+, and TaP^- for the non-anomalous entities as:

$$TaP^+(E,k) = \frac{1}{|E^+|}\sum_{e_i \in E^+} TaP(e_i,k)$$

$$TaP^-(E,k) = \frac{1}{|E^-|}\sum_{e_i \in E^-} TaP(e_i,k)$$

Both values are combined in a *Time aware Precision* variant denoted as TaP_α and defined as:

$$TaP_\alpha(e_i,k) = \alpha \cdot TaP^+(e_i,k) + (1-\alpha) \cdot TaP^-(e_i,k)$$

where the parameter α controls the impact of anomalous and non-anomalous cases in the final score. This allows to control the balance between cases and the representation of each class on the final result. Again, this parameter is problem related and its value will depend on each specific situation. In this sense, $\alpha = 0.5$ will balance both cases, but other approaches could be considered. For example, with $\alpha = 1$ only positive (i.e. anomalous) cases would be taken into account and non-anomalous cases will not be considered, which could be interesting when studying the early detection of a disease where non infected cases are non important.

TABLE 1. Summary of the main characteristics of the datasets used in the evaluation.

Dataset	*Entities & items*	*Normal*	*Anomalous*	*Total*
Depression	Entities	752	135	887
	Items	481,837	49,557	531,394
	Items per entity	640.7	367.1	599.1
Network attacks	Entities	10,045	65,655	75,700
	Items	1,566,602	131,249	1,697,851
	Items per entity	155.96	1.99	22.43
Floods	Entities	66,812	1,454	68,266
	Items	537,932	25,630	563,562
	Items per entity	8.05	17.63	8.25

IV. EXPERIMENTAL SETTINGS

A. DATASETS

Three datasets coming from three different and independent domains are used for evaluation purposes: a dataset for the detection of depression based on social networks, a dataset used for the detection of computer network attacks and a dataset of weather observations data for the detection of floods.

The following subsections provide the details for each one of them and how the early detection problem is defined in each case.

1) DEPRESSION DATASET

This dataset has been specifically gathered for the Workshop on Early detection prediction on the Internet 2017 (eRisk) [6] and contains public posts from Reddit published by individuals which have been manually tagged as depressed or non-depressed based on self-reports of diagnosed depression. The dataset contains posts for a period of about a year for each user.

In this case, the set of entities E corresponds to the different subjects considered in the dataset, while the sequence of items for each entity is represented by the subject's posts.

Table 1 shows a brief summary of the main numbers associated with the datasets considered. This dataset constitutes the smallest one, with just 887 entities and slightly more than half a million items. Also interestingly, the average number of items per entity is higher than the other datasets. In the experiments the features defined in [25] and [26] will be utilised, but limited to those considering individual post characteristics and not taking into account aggregated features for a sequence of posts.

2) NETWORK ATTACKS DATASET

This dataset includes data traffic from a video surveillance network where different attacks are performed throughout several days and is used to test Network Intrusion Detection Systems (NIDS) [27]. In our case, we focus on the OS Scan Attack from this dataset that scans the network for hosts and their operating systems to reveal potential vulnerabilities.

In order to model this as an early detection problem we have created bidirectional flows from the communication network data [28]. In this way, the set of entities, E, corresponds to data flows, where each flow is composed of a sequence of packets (i.e. items) defined by the same pairs source IP address - destination IP address, source port - destination port and protocol. Furthermore, to improve the flow division the timestamp of data flow packets is also considered, setting a threshold for the time between two consecutive packets to 0.1 seconds and the threshold for the flow duration to 1 second [20].

In total, more than 75 thousand entities and nearly 1.7 million different items are being considered (Table 1). For evaluation purposes, the features proposed in [27] are employed, which consider only individual characteristics from each packet, disregarding aggregated flow features.

3) FLOODS DATASET

Using the data provided by the National Center for Environmental Information (NCEI) of the National Oceanic and Atmospheric Administration (NOAA) from the USA we have constructed a floods dataset for 2018 at the United States.

Based on the Storm Events Database [29] provided by the NCEI, we have collected the details about timing and location for all flood events produced in 2018. Moreover, the Integrated Surface Dataset (ISD) from the NCEI provides worldwide surface hourly weather observations, including atmospheric pressure, temperature, dew point, atmospheric winds and precipitations, among others [30]. For each flood, we have identified the closest meteorological station and have collected data for the whole year, including the period of time when the flood occurred.

In order to study the early detection of floods, we have converted the dataset to a series of raining sequences. A raining sequence starts when a precipitation observation is obtained and will continue until the precipitation stops. A limit of up to 2 hours without any precipitation has been set for the sequence to finish. Then, a raining sequence is classified as flood if the flood occurred during the raining sequence or up to 24 hours.

In this case, the set of entities, E, corresponds to the raining sequences and precipitation observations constitute the sequence of items for each entity. The experiments will be performed considering as features the different weather observation data collected by the NCEI in each meteorological station.

From Table 1 we can observe how this dataset has a relatively high number of entities, close to 70 thousand and the number of items is above 500 thousand. Probably, the most relevant feature of this dataset is that it is mainly unbalanced, with just 2.1% of entities considered anomalous (i.e. floods).

B. EVALUATION PROTOCOLS

For evaluation purposes, the set of entities, E, will be divided into 5 folds and each one of them into two non-overlapping sets: training and testing. We will present the mean result of this experiments and we use a two-tail paired t-test with a $p - value < 0.05$ to indicate the significance of the performance differences. The literature collects mainly two types of evaluation on different early detection problems: batch or streaming [6], [24].

In the batch evaluation the sequence of items, I_e, is divided into various homogeneous and consecutive batches, and each one is processed independently. Therefore, for each entity e, $I_e = (< I_1^e, I_1^e >, < I_2^e, t_2^e >, \ldots, < I_n^e, t_n^e >)$ is split into B batches, where batch j, B_j, is defined as the sequence of items occurring between time $[(j-1)\cdot n/B + 1]$ and $[j \cdot n/B]$:

$$B_j = \left(< I_{\left[\frac{(j-1)\cdot n}{B+1}\right]}, t_{\left[\frac{j-1)\cdot n}{B+1}\right]} >, \ldots, < I_{\left[\frac{j\cdot n}{B}\right]}, t_{\left[\frac{j\cdot n}{B}\right]} > \right)$$

Since entities in E are independent and can have different lengths, batches will be homogeneous for each entity e, but batches may have different sizes for each entity.

Each batch is processed individually and, typically, all previous batches can be considered by the early detection model. Therefore, the function $f(l_e, B_j, \left[\frac{j\cdot n}{B}\right]), j \in [1, B]$ must be processed until a final output is obtained or until B is reached.

In our experiments, following previous works [6], [26], we have considered $B = 10$ for the test-set, with each batch containing 10% of the items.

In streaming evaluation the sequence of items, I_e, is processed individual and sequentially. Consequently, for each entity e, the function $f(l_e, I_e, k), k \in [1, n]$ must be processed, until a final output is obtained or until n is reached. As in the previous case, when processing item k, I_k, all previous items, $I_1, \ldots, I_{k-1}$, can be taken into account by the model.

It is interesting to note that, in a batch evaluation, when a final result is provided, the exact batch used is identified (e.g. B_j). However, the exact item from the batch used to produce this decision can not be identified and, therefore, for evaluation purposes, the last item of the batch will be considered (e.g. $I_{\left[\frac{j\cdot n}{B}\right]}$). On the other side, in streaming evaluation, the exact item, I_k, used to reach a final decision is certainly determined.

C. EVALUATION METRICS

For all experiments we report results on *ERDE* and $F_{latency}$ as well as *TaP*.

The parameter o, used in *ERDE* and *TaP* to set the penalty point, is set for each dataset considering two cases, low and high penalty points:

- Depression dataset: $o = 5$ for low penalty point and $o = 50$ for high penalty point.
- Network attacks dataset: $o = 1$ and $o = 10$, respectively for low and high.
- Floods dataset: low penalty point set at $o = 2$ and high penalty point set at $o = 20$.

The parameter p for $F_{latency}$ is computed for each dataset to obtain 50% latency penalty at the median number of items.

The remaining parameters for *TaP*, namely λ and α, are explored in the experiments section (Section V) and different values are considered.

D. MODELS

Initially, to test the behaviour of the different evaluation metrics, we present five synthetic baseline models that are used to represent extreme cases in the evaluation. For each one of them labels are assigned, considering original labels, as follows:

- $Oracle_n$: produces delays before item n for all entities and then the correct prediction for each entity is generated. Therefore, an $Oracle_1$ would represent a best-case scenario where all correct predictions are produced after processing the first item of each entity.
- $Elcaro_n$: works as an inverse *Oracle*, delaying the prediction before item n and then providing the wrong prediction for each entity. In this case, any *Elcaro* would represent a worst-case scenario, independently of the time required to generate the prediction.
- $Positive_n$: a delay is produced before item n and then all entities are tagged as positive (i.e. anomalous) cases.
- $Negative_n$: in this case, after item n, all entities are predicted as negative (i.e. normal or non-anomalous).
- *Random*: the three possible outputs (positive, negative or delay) are generated randomly with equal probabilities.

Moreover, in the experimental evaluation we will consider the following off-the-shelf machine learning algorithms. The *scikit-learn* [31] implementation will be used to generate basic models in order to measure their performance and evaluate the metrics behaviour. Each one of the models include the parameters used for training:

- LinearSVC:
  - C = 1
  - class_weight = 'balanced'
  - dual = False
  - max_iter = 1000
- ExtraTree:
  - n_estimators = 50
  - bootstrap = False
  - class_weight = None
- AdaBoost:
  - n_estimators = 1000
  - learning_rate = 2.0
  - algorithm = 'SAMME.R'
- Random Forest:
  - n_estimators = 500
  - class_weight = None
  - max_features = 'sqrt'
  - max_depth = 7
  - bootstrap = False
- Logistic Regression:
  - C = 0.1
  - class_weight = 'balanced'
  - dual = False
  - penalty = 'l2'
  - solver = 'sag'

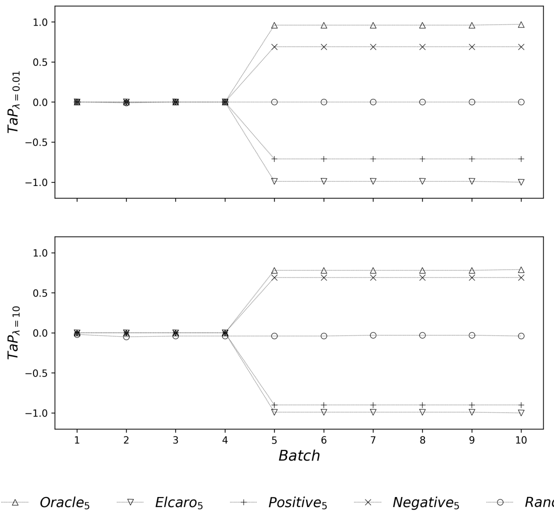


FIGURE 2. ***$TaP_{o=5}$*** **for Depression dataset with $\lambda = 0.01$ in the higher figure and $\lambda = 10$ in the lower figure.** ***$Oracle_5$*, *$Elcaro_5$*, *$Positive_5$*, *$Negative_5$*** **and** ***Random*** **models are represented in each figure.**

This selection of models was made based on the work presented in [32]. For all algorithms a model for each point of measure has been defined as it is more thoroughly explained in the following section.

V. EXPERIMENTS

In this section we present the results for our experiments, which we divide in two blocks: batch and streaming evaluation. For the batch evaluation protocol we have split each dataset into 10 batches, each one with 10% of the items for each entity. In the streaming evaluation protocol, on the contrary, entities' items are processed individual and sequentially until a final decision is obtained.

A. BATCH EVALUATION

Firstly, we focus on *TaP* metric and the parameter λ using the Depression dataset. We compute $TaP_{o=5}$ for all baselines and we test different values of λ (i.e. 0, 0.01, 0.1, 1 and 10). In Figure 2 we present results for $\lambda = 0.01$ and $\lambda = 10$ as representative of the parameter operation. This will introduce a smooth but noticeable penalty for the first value and a severe one for the second, as shown in Figure 1.

In both cases, as expected, $Oracle_5$ and $Elcaro_5$ take the higher and lower values, respectively. However, when $\lambda = 10$, $Oracle_5$ is not able to reach a perfect score. This is due to the fact that some entities (i.e. subjects, in this case) required more than 5 items to make the correct prediction, as a result of the batch distribution and that the penalisation introduced by $\lambda = 10$ is high. Meanwhile, the smother penalisation introduced with $\lambda = 0.01$ allows $Oracle_5$ to reach almost a perfect score. It is interesting to observe how $Positive_5$ and $Negative_5$ are closer to $Oracle_5$ and $Elcaro_5$, respectively, with $\lambda = 10$ since the penalisation for a late detection is, almost, equivalent to a wrong prediction. Also, the symmetry shown on this figures is due to the synthetic nature of the models.

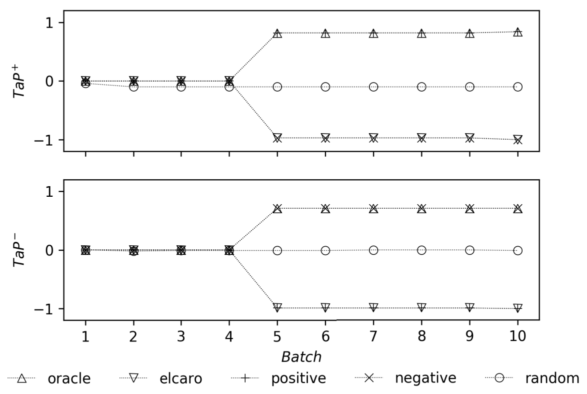


FIGURE 3. ***$TaP^+_{o=5}$*** **(up) and** ***$TaP^-_{o=5}$*** **(down) for Depression dataset with $\lambda = 0.01$.** ***$Oracle_5$*, *$Elcaro_5$*, *$Positive_5$*, *$Negative_5$*** **and** ***Random*** **models are represented in each figure.**

In the next set of experiments, we examine the behaviour of TaP_α. For this purpose, we fix the value of $\lambda = 0.01$, as this value introduces a slight but noticeable penalisation for all datasets in this form of evaluation, and we show how TaP^+ and TaP^- behave independently in Figure 3.

From the figure, we observe how each measure focuses only on positive and negative cases, respectively. Therefore, on TaP^+ $Oracle_5$ and $Positive_5$ models achieve equal and the highest scores, meanwhile on TaP^- $Oracle_5$ and $Negative_5$ achieve the highest scores. Alternatively, $Negative_5$ obtains a -1 score on TaP^+ as all positive cases are predicted as negative, and an equivalent response is obtained for $Positive_5$ on TaP^-.

Next, we study the effect of α parameter for TaP_α. For this purpose, we calculate $TaP_{o=5}$ for all baselines using the same dataset and testing different values of α: 0.5, 0.75, 0.9 and 0.95. Figure 4 shows the results obtained.

It is interesting to note how α does not have an effect on the performance of $Oracle_5$ and $Elcaro_5$ models, since, in the former, there are no wrong predictions and, in the latter, all positives cases are wrongly predicted. However, the effect of α is clearly observed on the $Positive_5$ and $Negative_5$ models. As α increases, more importance is provided to only positive predictions and, therefore, the performance of $Positive_5$ model becomes closer to $Oracle_5$ and, correspondingly, $Negative_5$ gets closer to $Elcaro_5$.

In the remaining experiments, although stated otherwise, the parameters for *TaP* are fixed to: $o = 5$, $\lambda = 0.01$ and $\alpha = 0.90$.

Finally, we compare the performance of the different metrics (i.e. *ERDE*, $F_{latency}$ and *TaP*) for the baseline methods using the three datasets. We provide two outputs for the *Oracle* model, at 1 and 5 items, respectively. Also, we use two values for the parameter o for *ERDE* and *TaP* for each dataset to capture low and high penalty points. For the Depression dataset we consider $o = 5$ and $o = 50$, respectively, following [6]; for the Network attacks dataset we set $o = 1$ [20] and $o = 10$ and Floods dataset uses $o = 2$ and $o = 20$. Table 2 summarises the results obtained.

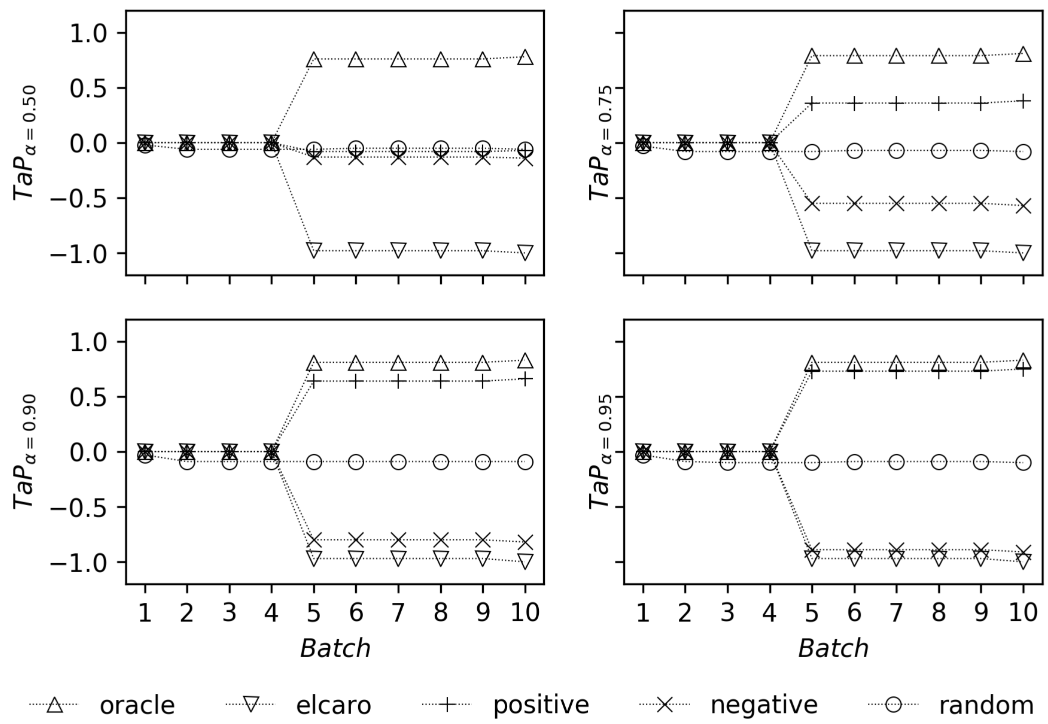


FIGURE 4. ***TaP_α* for Depression dataset with $\alpha = 0.5$ (up left), $\alpha = 0.75$ (up right), $\alpha = 0.9$ (down left) and $\alpha = 0.95$ (down right). Parameters *o* and λ are fixed at 5 and 0.01, respectively. *Oracle*$_5$, *Elcaro*$_5$, *Positive*$_5$, *Negative*$_5$ and *Random* models are represented on each figure.**

Regarding the *ERDE* metric, we can observe the difficulty of interpretation and comparability among datasets. $Oracle_1$ and $Oracle_5$ were expected to reach almost perfect scores in all cases, which is mainly true to the Depression and Floods datasets where $ERDE_{o=high}$ even reach 0.0. However, on the Network attacks dataset, both models obtain a relatively low score (0.4337 and 0.6338). This is due to the small number of items in most positive entities in this dataset, approximately 2 packets for anomalous flow, which introduces a high penalization for correct predictions.

Focusing on $F_{latency}$, it is interesting to note that this is a final metric and, therefore, just one score is provided since it is not possible to measure the performance at different points (i.e. parameter o). Although the minimum value is obtained for *Elcaro* and *Negative* models, and that $Oracle_1$ obtains the maximum value, the penalization introduced for the model $Oracle_5$ must be noticed. This is motivated because both models achieve a perfect $F1$ measure and the latency component penalise the differences in the prediction moment. In particular, in the case of Network Attacks or Floods datasets, where the number of items per entity is smaller, the penalization introduced is so that even when the number of items, k, is 5 the measure obtains half the value or less.

On the other side, *TaP* is able to provide the maximum scores, or close to, for *Oracle* models and minimum scores for *Elcaro* on all datasets. Also, values are comparable and same models obtain similar scores on different datasets. Batch evaluation may produce, in some cases (i.e. for $Oracle_1$ in the Depression dataset and $o = low$), that the maximum score is not achieved. This is because the last item of the batch is considered for the metric calculation and may introduce a small decrement. Interestingly, this highlights the difference between $Oracle_1$ and $Oracle_5$ in the Depression dataset, where the slightly worse performance of the latter is patent from the *TaP* scores obtained.

Next we present the results for standard Machine Learning models using the same three datasets and metrics. To do so, five different algorithms had been trained with two different points of prediction, on batches 1 and 5. The same parameters are used for all three datasets and points defined. These results are shown in Table 3.

The variation observed between datasets shows that with the previously selected parameters for the metrics the difference between $o = low$ and $o = high$ is slightly higher for *TaP* than for *ERDE* when the systems show a bad performance. There are bigger differences for *ERDE* between $o = low$ and $o = high$ on the better performing models, but the changes between models are smaller which could make more difficult selecting the better model. That can be seen for Network Attacks dataset for $ExtraTree_1$ and $AdaBoost_1$ where *ERDE* achieve 0.4337 for both models while the other metrics show different values. In particular, *TaP* obtain 0.9858 and 0.9857, showing a mild improvement in terms of definition.

TABLE 2. Performance for baseline models on batch evaluation. *TaP* uses $\lambda = 0.01$ and $\alpha = 0.90$. Parameter *o* is set to two values, $(low, high)$, for *ERDE* and *TaP* in each dataset: $(5, 50)$ for Depression, $(1, 10)$ for Network Attacks and $(2, 20)$ for Floods.

Dataset	Model	$ERDE_{o=low}$	$ERDE_{o=high}$	$F_{latency}$	$TaP_{o=low}$	$TaP_{o=high}$
Depression	$Oracle_1$	0.047	0.0	1.0	0.9849	1.0
	$Oracle_5$	0.1002	0.0	0.9227	0.9814	1.0
	$Elcaro_1$	0.2812	0.2812	0.0	-1.0	-1.0
	$Positive_1$	0.176	0.129	0.2642	0.7883	0.8
	$Negative_1$	0.1522	0.1522	0.0	-0.8033	-0.8
	$Random_1$	0.1699	0.1361	0.2376	0.0212	0.0413
Network Attacks	$Oracle_1$	0.4337	0.0001	1.0	0.9858	0.9939
	$Oracle_5$	0.6338	0.0003	0.5002	0.9768	0.9939
	$Elcaro_1$	0.9824	0.9824	0.0	-1.0	-1.0
	$Positive_1$	0.5487	0.1152	0.9289	0.8	0.8
	$Negative_1$	0.8673	0.8673	0.0	-0.8142	-0.8061
	$Random_1$	0.7088	0.4921	0.6353	-0.0091	-0.0083
Floods	$Oracle_1$	0.0087	0.0	1.0	0.9947	0.9997
	$Oracle_5$	0.0177	0.0	0.3757	0.9729	0.9997
	$Elcaro_1$	0.0421	0.0421	0.0	-1.0	-1.0
	$Positive_1$	0.0295	0.0209	0.0417	0.7948	0.7997
	$Negative_1$	0.0213	0.0213	0.0	-0.8001	-0.8
	$Random_1$	0.0266	0.0217	0.043	0.0029	-0.0167

In the Oracle evaluation, the best results were expected to be obtained by using a higher o (i.e. $o = high$) as *TaP* parameter. That will imply a lower penalty in the evaluation and that is what can be observed in the Machine Learning models evaluation as well.

As shown in the Oracle evaluation, better results were expected to be obtained the sooner the decision was taken. This might not be the same situation regarding Machine Learning models as some systems could perform better with less information than others or also in some cases it would be possible to get worst results with more information.

If we analyse the results for Network Attacks dataset it can be seen that all the models performance improve the sooner the decision is taken. One exception to this behaviour can be observed in Floods dataset results, where $LinearSVC_1$ and $LinearSVC_5$, for example, achieve almost the same values for every metric. Also, as explained in the baseline models evaluation, better results are achieved for a higher point of penalty regarding the proposed metric.

Finally, the differences shown in the behaviour of $F_{latency}$ metric and *TaP* between the baseline models and Machine learning models evaluation must be noted.

B. STREAMING EVALUATION

In the streaming evaluation, instead of batches, each item for each entity is processed and evaluated sequential and individually.

Since the gap between two consecutive measures, in this case, is just one item, we have decided to set $\lambda = 0.1$ to observe variations in the metric more clearly. In contrast with a batch evaluation where the gap between two consecutive batches (and measures) would typically be of quite a few items, a value of $\lambda = 0.01$ was selected.

Before delving into the experiments, we present in Figure 5 the results for the baseline models in the Depression dataset. Observe that the X-axis represents the number of items used

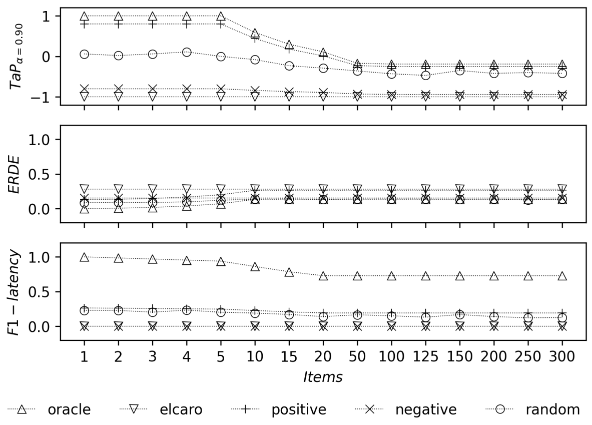


FIGURE 5. *TaP* (top), *ERDE* (middle), $F_{latency}$ (bottom) for Depression dataset with $o = 5$, $\lambda = 0.1$ and $\alpha = 0.9$. Baseline models, *Oracle*, *Elcaro*, *Positive*, *Negative* and *Random*, are showed on each figure. The X-axis represents the number of items used by the baseline models to compute the predictions for all entities.

by a baseline model to produce its prediction, and the Y-axis represents the score obtained. Therefore, *Oracle* at $x = 10$ is an $Oracle_{10}$ and the *TaP* score achieved is approximately 0.5.

Regarding the *TaP* metric for *Oracle* and *Positive* on Figure 5, we can observe how, after the penalisation point of $o = 5$, there is a linear decrease in the performance, as expected since the predictions, although correct, are generated late.

In the same figure 5, *ERDE* shows very little difference between the five models which decrease the ability to differentiate among models performance.

Lastly, $F_{latency}$ starts with a close to perfect value in the first measurement points for *Oracle* model, and it stabilises around 20 items. As it can be seen, *Oracle* model presents a big difference with the rest of the models, which again cannot be easily differentiated.

Focusing more on *TaP* we explore the effect of λ on Figure 6 setting $\lambda = 10$ for $TaP_{o=5}$ and $TaP_{o=50}$, again in

TABLE 3. **Performance for Machine Learning models on batch evaluation. *TaP* uses $\lambda = 0.01$ and $\alpha = 0.90$. Parameter *o* is set to two values, (*low*, *high*), for *ERDE* and *TaP* in each dataset: (5, 50) for Depression, (1, 10) for Network Attacks and (2, 20) for Floods.**

Dataset	Model	$ERDE_{o=low}$	$ERDE_{o=high}$	$F_{latency}$	$TaP_{o=low}$	$TaP_{o=high}$
Depression	$AdaBoost_1$	0.1407	0.1097	0.3096	0.2762	0.2855
	$AdaBoost_5$	0.1737	0.1081	0.3016	0.2878	0.2994
	$ExtraTree_1$	0.1458	0.1457	0.0929	-0.7121	-0.7088
	$ExtraTree_5$	0.1497	0.149	0.0378	-0.764	-0.7603
	$LinearSVC_1$	0.1513	0.1513	0.0141	-0.7903	-0.7869
	$LinearSVC_5$	0.1522	0.1522	0.0	-0.8037	-0.8
	$LogisticRegression_1$	0.1298	0.0951	0.363	0.354	0.366
	$LogisticRegression_5$	0.1537	0.0776	0.394	0.5177	0.5322
	$RandomForest_1$	0.1489	0.1488	0.0414	-0.7633	-0.76
	$RandomForest_5$	0.1511	0.1511	0.0132	-0.7904	-0.7867
Network Attacks	$AdaBoost_1$	0.4337	0.0001	1.0	0.9857	0.9939
	$AdaBoost_5$	0.6338	0.0003	0.5002	0.9767	0.9939
	$ExtraTree_1$	0.4337	0.0001	1.0	0.9858	0.9939
	$ExtraTree_5$	0.6338	0.0003	0.5002	0.9768	0.9939
	$LinearSVC_1$	0.4393	0.0113	0.9935	0.9625	0.9707
	$LinearSVC_5$	0.637	0.0115	0.497	0.9536	0.9706
	$LogisticRegression_1$	0.4403	0.0068	0.9956	0.9743	0.9824
	$LogisticRegression_5$	0.6405	0.0069	0.498	0.9653	0.9824
	$RandomForest_1$	0.4337	0.0001	1.0	0.9857	0.9939
	$RandomForest_5$	0.6338	0.0003	0.5002	0.9767	0.9939
Floods	$AdaBoost_1$	0.0255	0.0192	0.0386	0.3863	0.39
	$AdaBoost_5$	0.0322	0.0192	0.0146	0.3703	0.3893
	$ExtraTree_1$	0.0213	0.0213	0.0063	-0.7943	-0.7942
	$ExtraTree_5$	0.0213	0.0212	0.003	-0.7946	-0.7927
	$LinearSVC_1$	0.0213	0.0213	0.0	-0.8001	-0.8
	$LinearSVC_5$	0.0213	0.0213	0.0	-0.8018	-0.8
	$LogisticRegression_1$	0.0246	0.0206	0.0423	-0.0625	-0.0601
	$LogisticRegression_5$	0.0277	0.0196	0.0179	-0.0907	-0.0778
	$RandomForest_1$	0.0213	0.0213	0.0	-0.8001	-0.8
	$RandomForest_5$	0.0213	0.0213	0.0	-0.8018	-0.8

the Depression dataset. As expected, there is an important decrease in the performance of *Oracle* and *Positive* after the penalization point, but there is also an impact on the final score. In this sense, $Oracle_{10}$ and subsequent achieve values close to -1 for $TaP_{o=5}$, while for $TaP_{o=50}$ $Oracle_{100}$ and next are decreased but only to around 0.25. This is due to the effect of the logistic function in the penalty, that is higher for lower values of o.

Taking into account the results of baseline models evaluation, we proceed to show the results for the Machine Learning algorithms selected and already used in the batch evaluation section.

Figures 7,8 and 9 show the behaviour of *ERDE*, $F_{latency}$ compared to $TaP_{\alpha=0.90}$. Both for *ERDE* and *TaP* the o value selected was the $o = low$ used previously. That is, $o = 5$ for Depression, $o = 1$ for Network Attacks and $o = 2$ for Floods dataset. Also, λ parameter is defined to 0.1 in order to show a better impact of the penalisation when the difference between the number of elements taken into account is small as it was introduced at the beginning of this section. This happens, for example with the first values of the graphics as they are closer together than the rest.

Due to the performance of the algorithms and the distribution of the measurement points, the metrics output is almost completely stable. The exceptions can be located in three different cases with different explanations. First, in the case of Figure 8, where Network Attacks dataset results are

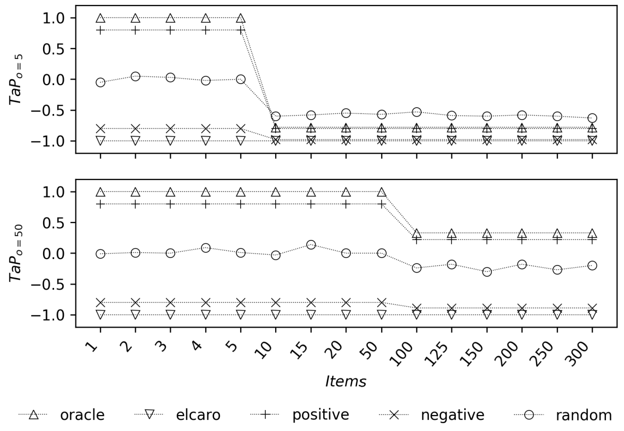


FIGURE 6. ***TaP* penalisation for streaming evaluation over Depression dataset with $\lambda = 10$ and $\alpha = 0.90$. *TaP* with $o = 5$ (top) and $o = 50$ (bottom) for Depression dataset. Parameters λ and α are fixed to 10 and 0.9, respectively. Baseline models, *Oracle*, *Elcaro*, *Positive*, *Negative* and *Random*, are showed on each figure. The X-axis represents the number of items used by the baseline models to compute the predictions for all entities.**

presented, a drastic change can be observed in all metrics, but specially in *ERDE* and $F_{latency}$ as the great majority of positive entities have around 2 items and that is where the penalty for $o = low$ is introduced. Secondly, we can focus on Depression dataset, showed in Figure 7, where we can observe how *RandomForest* and *ExtraTree* models show a

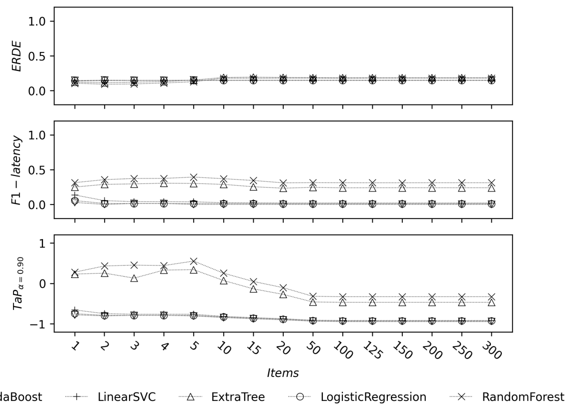


FIGURE 7. Comparison between *TaP*, *ERDE* and *F1 – latency* for streaming evaluation over Depression dataset with $\lambda = 0.1$ and $\alpha = 0.90$. Machine Learning selected models are showed in each figure. The X-axis represents the number of items used by the baseline models to compute the predictions for all entities.

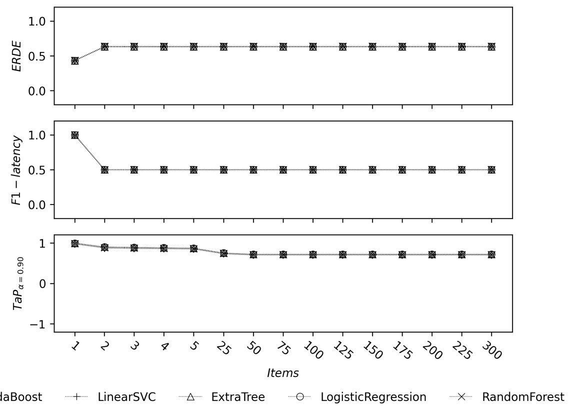


FIGURE 8. Comparison between *TaP*, *ERDE* and *F1 – latency* for streaming evaluation over Network Attacks dataset with $\lambda = 0.1$ and $\alpha = 0.90$. Machine Learning selected models are showed in each figure. The X-axis represents the number of items used by the baseline models to compute the predictions for all entities.

better performance, for $F_{latency}$ and *TaP*, achieving its best value at point 5. In this case, *ERDE* is not capable of clearly differentiate between those models. Lastly, in the results for Floods dataset, shown in Figure 9, *ERDE* and $F_{latency}$ are not capable to display the differences when the model obtain poor results and just *TaP* is able to do that for *ExtraTree* and *RandomForest* models. This shows, as it was seen for baseline models, the higher granularity provided by *TaP* metric against *ERDE* and $F_{latency}$.

Lastly, if the results for *TaP* metric from each dataset and the penalisation points are taken into account, changes in the output value can be observed and explained. If we take Depression dataset as an example, an increase in the value of the metric can be seen up to point 5. This is explained by some algorithms taking better decisions after at least two elements have been considered. At the same time, this does not improve the final value as the penalty is increased for more than 5 items. After the penalisation we can see a stabilisation of the metric value. This is related to the x-axis not showing a proportion of the total number of items for each dataset, instead, a more natural approach to the time distribution is shown.

VI. DISCUSSION

Batch evaluation provides a more straightforward and less resource consuming approach, and it is appealing if entities are homogeneous. However, if entities are heterogeneous (i.e. different number of items) the evaluation may penalise entities with smaller sequences of items. On the other side, streaming evaluation is more resource demanding since a whole sequence of items for each entity may have to be processed, but a better evaluation granularity can be achieved since the exact item used for the final decision can be identified.

Moreover, some details about the tuning of the metric should be discussed as two parameters that have to be defined for each problem must be carefully selected. On the one hand

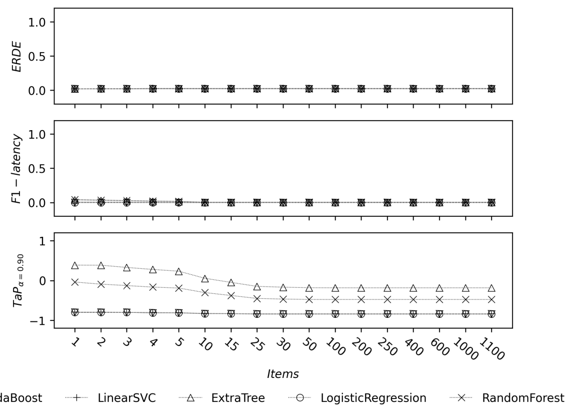


FIGURE 9. Comparison between *TaP*, *ERDE* and *F1 – latency* for streaming evaluation over Floods dataset with $\lambda = 0.1$ and $\alpha = 0.90$. Machine Learning selected models are showed in each figure. The X-axis represents the number of items used by the baseline models to compute the predictions for all entities.

the parameter o decides after what point the penalty is applied and this could be extracted from the problem itself and the urgency of the detection. On the other hand, α parameter defines how much False Positives and False Negatives affect the outcome and it also should be taken into account for better results.

Also, when parameters are discussed, we must notice that metrics as $F_{latency}$ require to know the whole dataset in order to compute the parameter p. When applying *ERDE* the amount of elements is required in advance to generate the penalisation factor. As opposed, in the case of *TaP*, where the parameters depend on the nature of the problem itself and not in the specific data of each dataset.

VII. CONCLUSION

In this paper, we presented the problem of early detection of anomalies over three datasets from different backgrounds. We can conclude that time aware metrics are relevant to properly evaluate time sensitive models. The chosen metrics must be able to be easily interpreted and to represent the performance as well as the promptness of response of the system. This is achieved by the development of the *TaP* metric which is able to provide a dataset agnostic way of measuring machine learning models in time aware environments.

In the future, we expect to extend this research in different ways. We would like to research specific early detection models for some of the problems considered in this work (e.g. network attacks) and analyse their performance using *TaP*. Within that work we will delve into feature extraction or generation, preprocessing and model definition. Also interesting is the use different datasets from the same topic to compare the performance of different machine learning models in a time-aware prediction, using the methodology and metric proposed.

REFERENCES

[1] V. Chandola, A. Banerjee, and V. Kumar, ''Anomaly detection: A survey,'' *ACM Comput. Surv.*, vol. 41, no. 3, p. 15, 2009.

[2] W. P. Pierskalla and D. J. Brailer, ''Applications of operations research in health care delivery,'' *Handbooks Oper. Res. Manage. Sci.*, vol. 6, pp. 469–505, 1994.

[3] R. Gouriveau, K. Medjaher, and N. Zerhouni, *From Prognostics and Health Systems Management to Predictive Maintenance 1: Monitoring and Prognostics*. Hoboken, NJ, USA: Wiley, 2016.

[4] R. K. Mobley, *An Introduction to Predictive Maintenance*. Amsterdam, The Netherlands: Elsevier, 2002.

[5] E. Hutchings, M. Cloppert, and R. Amin, *Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains*, vol. 1. New York, NY, USA: Academic, 2011.

[6] D. E. Losada and F. Crestani, ''A test collection for research on depression and language use,'' in *Proc. Int. Conf. Cross-Lang. Eval. Forum Eur. Lang.* Cham, Switzerland: Springer, 2016, pp. 28–39.

[7] D. E. Losada, F. Crestani, and J. Parapar, ''Erisk 2020: Self-harm and depression challenges,'' in *Advances in Information Retrieval*, J. M. Jose, E. Yilmaz, J. Magalhães, P. Castells, N. Ferro, M. J. Silva, and F. Martins, Eds. Cham, Switzerland: Springer, 2020, pp. 557–563.

[8] F. Sadeque, D. Xu, and S. Bethard, ''Measuring the latency of depression detection in social media,'' in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, Feb. 2018, pp. 495–503.

[9] N. S. Samghabadi, A. P. L. Monroy, and T. Solorio, ''Detecting early signs of cyberbullying in social media,'' in *Proc. 2nd Workshop Trolling, Aggression Cyberbullying*. Marseille, France: European Language Resources Association (ELRA), May 2020, pp. 144–149. [Online]. Available: https://www.aclweb.org/anthology/2020.trac-1.23

[10] M. F. López-Vizcaíno, F. J. Nóvoa, V. Carneiro, and F. Cacheda, ''Early detection of cyberbullying on social media networks,'' *Future Gener. Comput. Syst.*, vol. 118, pp. 219–229, May 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X21000157

[11] Z. Zhao, P. Resnick, and Q. Mei, ''Enquiring minds: Early detection of rumors in social media from enquiry posts,'' in *Proc. 24th Int. Conf. World Wide Web*, May 2015, pp. 1395–1405.

[12] X. Zhou, A. Jain, V. V. Phoha, and R. Zafarani, ''Fake news early detection: An interdisciplinary study,'' 2019, *arXiv:1904.11679*.

[13] S. Hosokawa, M. Enomoto, K. Matsumoto, and M. Takahashi, ''A prototype of cyber incident diagnosis mechanism for cyber attacks early recognition support system,'' in *Proc. 10th Asian Control Conf. (ASCC)*, May 2015, pp. 1–5.

[14] S. N. Narayanan, A. Ganesan, K. Joshi, T. Oates, A. Joshi, and T. Finin, ''Early detection of cybersecurity threats using collaborative cognition,'' in *Proc. IEEE 4th Int. Conf. Collaboration Internet Comput. (CIC)*, Oct. 2018, pp. 354–363.

[15] M. Pivarníková, P. Sokol, and T. Bajtoš, ''Early-stage detection of cyber attacks,'' *Information*, vol. 11, no. 12, p. 560, Nov. 2020. [Online]. Available: https://www.mdpi.com/2078-2489/11/12/560

[16] H. K. Kalutarage, C. Lee, S. A. Shaikh, and F. L. B. Sung, ''Towards an early warning system for network attacks using Bayesian inference,'' in *Proc. IEEE 2nd Int. Conf. Cyber Secur. Cloud Comput.*, Nov. 2015, pp. 399–404.

[17] C. C. Zou, L. Gao, W. Gong, and D. Towsley, ''Monitoring and early warning for internet worms,'' in *Proc. 10th ACM Conf. Comput. Commun. Secur. (CCS)*, 2003, pp. 190–199, doi: 10.1145/948109.948136.

[18] K. Bsufka, O. Kroll-Peters, and S. Albayrak, ''Intelligent network-based early warning systems,'' in *Proc. Int. Workshop Crit. Inf. Infrastructures Secur.* Cham, Switzerland: Springer, 2006, pp. 103–111.

[19] D. Fernandez, L. Vigoya, F. Cacheda, F. J. Novoa, M. F. Lopez-Vizcaino, and V. Carneiro, ''A practical application of a dataset analysis in an intrusion detection system,'' in *Proc. IEEE 17th Int. Symp. Netw. Comput. Appl. (NCA)*, Nov. 2018, pp. 1–5.

[20] M. Lopez-Vizcaino, F. J. Novoa, D. Fernandez, V. Carneiro, and F. Cacheda, ''Early intrusion detection for OS scan attacks,'' in *Proc. IEEE 18th Int. Symp. Netw. Comput. Appl. (NCA)*, Sep. 2019, pp. 209–213.

[21] C. Xu, H. Lin, Y. Wu, X. Guo, and W. Lin, ''An SDNFV-based DDoS defense technology for smart cities,'' *IEEE Access*, vol. 7, pp. 137856–137874, 2019.

[22] A. Privalov, V. Lukicheva, I. Kotenko, and I. Saenko, ''Method of early detection of cyber-attacks on telecommunication networks based on traffic analysis by extreme filtering,'' *Energies*, vol. 12, no. 24, pp. 1–14, Dec. 2019. [Online]. Available: https://ideas.repec.org/a/gam/jeners/v12y2019i24p4768-d297800.html

[23] M. Trotzek, S. Koitka, and C. M. Friedrich, ''Utilizing neural networks and linguistic metadata for early detection of depression indications in text sequences,'' *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 3, pp. 588–601, Mar. 2020.

[24] D. E. Losada, F. Crestani, and J. Parapar, ''Overview of erisk at clef 2019 early risk prediction on the internet (extended overview),'' in *Proc. Int. Conf. Cross-Language Eval. Forum Eur. Lang.* Cham, Switzerland: Springer, 2019, pp. 1–18.

[25] F. Cacheda, D. F. Iglesias, F. J. Nóvoa, and V. Carneiro, ''Analysis and experiments on early detection of depression,'' in *Proc. CLEF Working Notes*, vol. 2125, 2018, pp. 1–12.

[26] F. Cacheda, D. Fernandez, F. J. Novoa, and V. Carneiro, ''Early detection of depression: Social network analysis and random forest techniques,'' *J. Med. Internet Res.*, vol. 21, no. 6, Jun. 2019, Art. no. e12554.

[27] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, ''Kitsune: An ensemble of autoencoders for online network intrusion detection,'' 2018, *arXiv:1802.09089*.

[28] B. Trammell and E. Boschi, ''An introduction to IP flow information export (IPFIX),'' *IEEE Commun. Mag.*, vol. 49, no. 4, pp. 89–95, Apr. 2011.

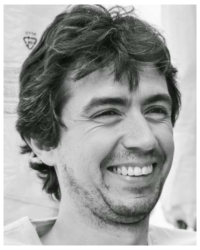
[29] NOAA. (2019). *Storm Events Database 2018*. [Online]. Available: Https://www.ncdc.noaa.gov/stormevents/

[30] NOAA. (2019). *Global Surface Hourly 2018*. [Online]. Available: https://catalog.data.gov/dataset/integrated-surface-global-hourly-data

[31] Scikit Learn. *Machine Learning in Python Scikit-Learn 1.1.2 Documentation*. Accessed: Oct. 4, 2022. [Online]. Available: https://scikit-learn.org/stable/

[32] R. I. Rafiq, H. Hosseinmardi, S. A. Mattson, R. Han, Q. Lv, and S. Mishra, ''Analysis and detection of labeled cyberbullying instances in vine, a video-based social network,'' *Social Netw. Anal. Mining*, vol. 6, no. 1, pp. 1–16, Dec. 2016.

MANUEL F. LÓPEZ-VIZCAÍNO was born in Lugo, Spain, in 1990. He received the B.S. degree in computer science from the University of A Coruña, Spain, in 2015, where he is currently pursuing the Ph.D. degree. He is currently working as an Assistant Teacher with the Department of Computer Sciences and Information Technologies of the University of A Coruña. His research interest includes the evaluation and application of early detection methods to anomalies in cybersecurity, although he is also interested in other topics regarding artificial intelligence, evaluation metrics, and network security.



FRANCISCO J. NOVOA was born in Ourense, Spain, in 1974. He received the M.S. degree in computer science from the University of Deusto, Spain, in 1998, and the Ph.D. degree in computer science from the University of A Coruña, Spain, in 2007. From 1998 to 2007, he developed his professional career in the business field of information technology, reaching multiple professional certifications such as CCNA, CCNP, and MCP. From 2007 to 2018, he was an Assistant Professor at the Computer Science Department, University of A Coruña. Since then, he has been an Associate Professor with the Computer Science Department. He is the author of 12 journal articles, ten book chapters, and more than 30 conference papers. His research interests include network security, intrusion detection, data flow analysis, the IoT, medical informatics, biomedical imaging, artificial intelligence, and neural networks.

DIEGO FERNÁNDEZ was born in A Coruña, Spain, in 1983. He received the Bachelor of Engineering degree in computer sciences and the Ph.D. degree in computer sciences from the University of A Coruña, Spain, in 2008 and in 2014, respectively. In 2008, he became a Research Assistant at the Telematics Laboratory. In 2014, he became an Assistant Professor at the Information and Communication Technologies Department. Since 2018, he has been an Associate Professor with the Computer Sciences Department. His research interests include information retrieval, collaborative filtering, and anomaly detection in computer networks.

FIDEL CACHEDA was born in Poissy, France, in 1973. He received the B.S., M.S., and Ph.D. degrees in computer science from the University of A Coruña, Spain, in 1994, 1996, and 2002, respectively.

From 1998 to 2006, he was an Assistant Professor at the Computer Science Department, University of A Coruña. Since then, he has been an Associate Professor with the Computer Science Department. He is the author of four books, nine book chapters, more than 20 journal articles, and more than 60 conference papers. His research interests include information retrieval, recommender systems, and early detection of anomalies applied to cybersecurity.